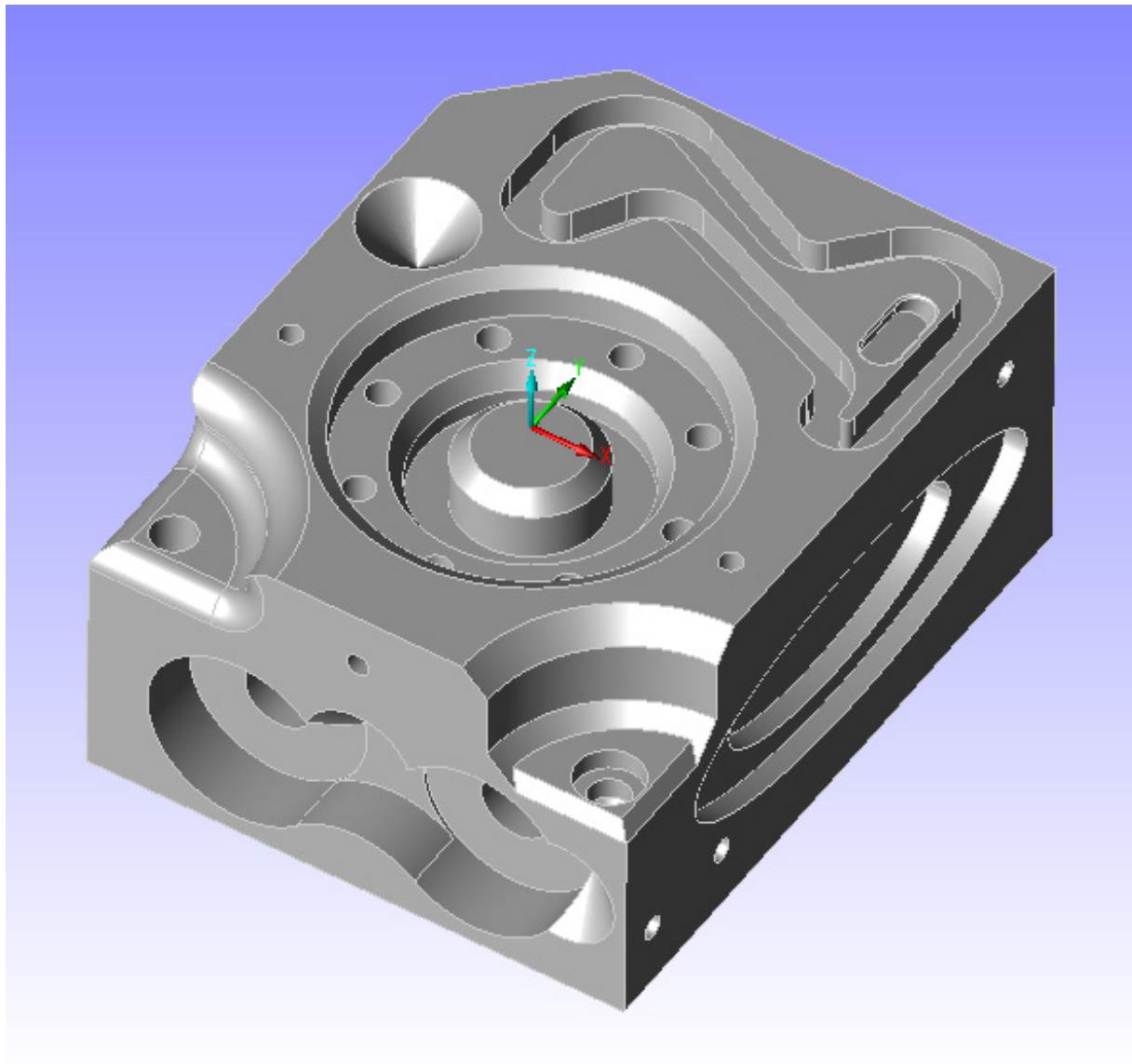


# Introduction to high level programming



© 2013 - 2016 Renishaw plc. All rights reserved.

Renishaw® is a registered trademark of Renishaw plc.

This document may not be copied or reproduced in whole or in part, or transferred to any other media or language, by any means, without the prior written permission of Renishaw.

The publication of material within this document does not imply freedom from the patent rights of Renishaw plc.

### **Disclaimer**

Considerable effort has been made to ensure that the contents of this document are free from inaccuracies and omissions. However, Renishaw makes no warranties with respect to the contents of this document and specifically disclaims any implied warranties. Renishaw reserves the right to make changes to this document and to the product described herein without obligation to notify any person of such changes.

### **Trademarks**

All brand names and product names used in this document are trade names, service marks, trademarks, or registered trademarks of their respective owners.

# **Introduction to high level programming**

## Care of equipment

Renishaw probes and associated systems are precision tools used for obtaining precise measurements and must therefore be treated with care.

## Changes to Renishaw products

Renishaw reserves the right to improve, change or modify its hardware or software without incurring any obligations to make changes to Renishaw equipment previously sold.

## Warranty

Renishaw plc warrants its equipment for a limited period (as set out in our Standard Terms and Conditions of Sale) provided that it is installed exactly as defined in associated Renishaw documentation.

Prior consent must be obtained from Renishaw if non-Renishaw equipment (e.g. interfaces and/or cabling) is to be used or substituted. Failure to comply with this will invalidate the Renishaw warranty.

Claims under warranty must be made from authorised service centres only, which may be advised by the supplier or distributor.

## Trademarks

Windows 98, Windows XP, Windows 2000 and Windows NT are registered tradenames of the Microsoft Corporation.

IBM is the tradename of the International Business Machines Inc

All trademarks and tradenames are acknowledged.

---

## Contents

1	Introduction to high level programming .....	6
1.1	Tutorial pre-requisites.....	6
1.2	Tutorial objectives.....	6
2	Introduction.....	7
3	Planning the program .....	8
4	Aligning the block .....	9
5	Variables.....	10
6	Variable naming convention - good discipline .....	11
7	Declaring variables.....	12
8	Create a prompt .....	14
8.1	Resulting prompt.....	16
8.2	Variable watch window .....	16
8.3	Assign the PCD size using inputs from a prompt.....	18
8.4	Select-Case.....	18
8.5	Assign the values in the 'Select-Case'.....	19
9	Assigning further variables .....	22
9.1	Concatenation of a file name .....	23
9.2	Assigning a value with a calculation.....	25
9.3	Assign the hole variables .....	25
10	Creating a loop .....	26
10.1	Add the hole measurement into the loop .....	28
10.2	Edit the hole nominal.....	29
10.3	Construct the PCD .....	31
10.4	Alternative loop method .....	31
11	Obtaining a feature value .....	32
11.1	Creating an IF statement.....	34
12	Device files .....	38
13	Creating folders and copying files .....	42
13.1	Creating folders.....	42
13.2	Copy file .....	43
13.3	Delete file .....	45

# **1 Introduction to high level programming**

## **1.1 Tutorial pre-requisites**

- Students should be familiar with the content of the basic tutorials

## **1.2 Tutorial objectives**

- To understand the fundamental principles of MODUS high level language
- To be able to declare, assign and use variables
- To be able to use loops to do repetitive tasks
- To be able to obtain a value from a previously measured feature
- To create an IF statement
- To be able to write data to a device (text file)
- To use a CALL statement to create a folder, copy files and delete files

## 2 Introduction

MODUS high level programming techniques are used to give DMIS programs additional capabilities. Most high level programming involves the use of one or more variables, conditional branching and file manipulation. A program that uses these techniques can take different execution paths depending on the current value of a variable.

Variables can be thought of as a container that holds data. Each type of data requires a different size container. Variables are used for counting, making decisions, changing text in reports and file names, holding the value of a calculation and many other uses.

This tutorial uses a simple PCD (bolt-circle) measurement program to illustrate how to use high level DMIS statements in a program. The numbers that define hole positions will be replaced with variables to generate generic code. This allows several different sizes of the same general work piece to be measured.

A prompt is used to ask the user what size they are measuring. Once a choice is made, the radius variables are set to the correct values and used to run the inspection routine regardless of the size chosen by the user. The actual diameter of the constructed circle will then be compared against the nominal value.

An IF statement will be used to determine which message will be displayed / printed to report based on error of the constructed diameter.

Finally, program commands will be used to create a new directory, copy the results file to the new directory then delete the original results file.

### 3 Planning the program

Planning is the one of the most important steps in writing any logical program. This type of pre-planning is similar to outlining a speech. Once the ideas have been written out, the rest becomes much more intuitive.

The following is a sample plan for this tutorial program using what programmers refer to as pseudo code. It replaces DMIS code with text statements, making it easy to think about the structure and ideas without using complicated code. Flow charting is also a good way to get the overall flow of the program and can save countless hours, reducing the need to rewrite programs.

- Align part to centre of PCD (section 4)
- Declare and assign variables (section 7)
- Create a loop to measure all the holes in a PCD pattern (section 10)
- Measure a hole inside the loop (section 10.1)
- Use a variable in place of the diameter value of the PCD, so the program can measure any diameter PCD (NOTE: For the purposes of this tutorial, only one option will be practically useful online) (section 11)
- Edit the hole name with a variable that combines the name and loop index variables (section 10.2)
- Use a variable for the rotational position of the hole within the measurement loop (section 10.2)
- End the loop (section 10.2)
- Construct the PCD with all the holes (section 10.3)
- Obtain the value of the diameter (section 11 )
- Determine the error of the PCD diameter (section 11.1)
- Create an IF statement to print a variety of error messages based on the error if over limit (section 11.1)
- Tolerance the PCD diameter (section 12)
- Create a folder to place the data (section 13.1)
- Copy the file to another folder (section 13.2)
- Delete the original file (section 13.3 )

Usually, there are details that are missed at this point, but it gives a good overview of the requirements that need to be addressed. With a basic plan in place, the programmer can fine tune as they progress, without losing focus of the overall goal.

## 4      **Aligning the block**

Before any programming commences in this tutorial, the student will need to have carried out a precise alignment of the training block, using the top face, the 80 mm diameter bore and a line constructed between the 80 mm diameter bore and one of the 7 mm diameter holes.

## 5 Variables

Each variable must be given a name, type and scope.

The variable type tells MODUS how much memory to set aside. Programming languages do this to make efficient use of memory. The following is a list of variable types that MODUS can use:

### 1. Integer – (Integer or Long Integer)

This is used for counting. The programmer would use an integer variable to hold the number of points in a scan or to count the number of times a loop has been run for example. The value 567 is an integer (extremely large integers will require a Long Integer type).

### 2. Decimal – (Double or Real)

This is typically used for numbers that define the position (XYZ or polar radius and angle), orientation (IJK vector) or size of a measurement feature. It can be used for any number that has a decimal point. The value 18.432 is a decimal value. A double can contain a value much larger than a real. If unsure, use a double. Double variables will be used throughout this tutorial.

### 3. String – (Character)

A 'String' variable is a set of alphanumeric characters or a sentence of information. Typically, this would be used to hold the operator name, file and path data and anything else that requires text information. The string "C:\Renishaw\Programs\Training Programs\" is an example of a string.

### 4. Boolean –

A 'Boolean' variable is a "Yes / No" or a "True / False" value; it is a Logic variable.

Using the wrong type of variable can produce incorrect results. For example, if an integer variable is declared, but assigned a decimal, the data to the right of the decimal will be lost since the memory size is not large enough to hold it.

The scope of a variable indicates the length of time in which a variable will be accessible for use .

1. **Local (module)** - A local variable is available only in the current program. Once the ENDFIL has been executed and program closed, a local variable will be erased and will be no longer available for use. This is true whether it is in a main program or subroutine.
2. **Global (program)** - A global variable is available for use in the program in which it was created as well as its subroutines. If a global variable is declared in a subroutine, it will also be available in the main program from which it was called. Once the main program ends, all global variables will be deleted from computer memory.
3. **Common (system)** - A common variable is permanently accessible unless it is specifically erased using the System variables selection in the tools menu of MODUS. Common variables are useful if the variable must be used in other programs. For example, a common (system) scope could be used to count the number of times a program has been run. It is best to use common variables sparingly if possible since you may end up with a very large list over time.

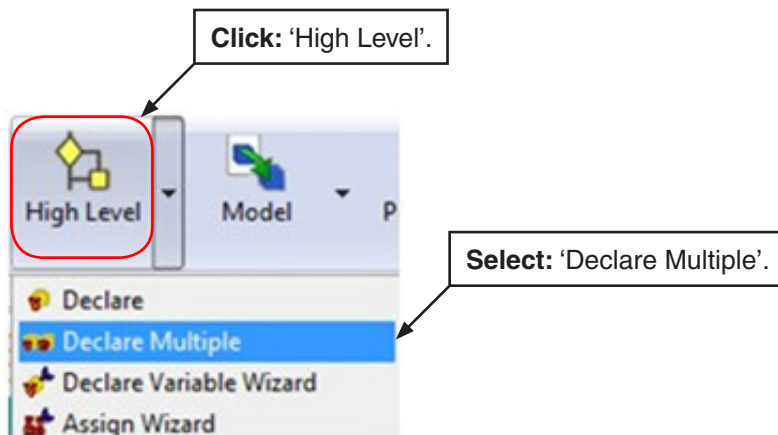
## 6 Variable naming convention - good discipline

Variable names can be called whatever a user decides (as long as they do not start with a number), but it is good practice within a company to standardise variable names. This tutorial uses a standard naming convention. Using a standard convention allows anyone reading through the program to know the scope, type and use of the variable.

Example: L\_STR\_MYVAR

- A. The first letter indicates the scope -
  - 1. L = Local
  - 2. G = Global
  - 3. C = Common
- B. The second segment of the name indicates the variable type -
  - 1. INT = Integer (for counting)
  - 2. DBL = Double (decimal values)
  - 3. STR = String (alphanumeric characters)
  - 4. BOO = Boolean (Yes / No or True / False)
- C. The last segment in the variable name should indicate what the variable is used for.

## 7 Declaring variables



Fill in the name, type and scope of the required variables listed below and then click 'OK'.

Integer or long integer (for the variables listed below use integer):

**L\_INT\_LOOP\_COUNT** - This controls the number of loops

**L\_INT\_PART\_CHOICE** - This will be assigned in a prompt window.

Real or double (for the variables listed below use double):

**L\_DBL\_BC\_DIA\_NOM** - Nominal diameter of the PCD

**L\_DBL\_BC\_DIA\_ACT** - Actual diameter of the PCD

**L\_DBL\_BC\_RAD** - Nominal Radius of PCD

**L\_DBL\_POL\_ANG** - Nominal polar angle of the PCD holes

**L\_DBL\_POL\_RAD** - Nominal polar radius of the PCD holes

**L\_DBL\_ERROR\_DIA** - The actual error in the PCD

Character or string:

**L\_STR\_FEAT\_NAME** - Used to name the feature names of the holes in the pattern

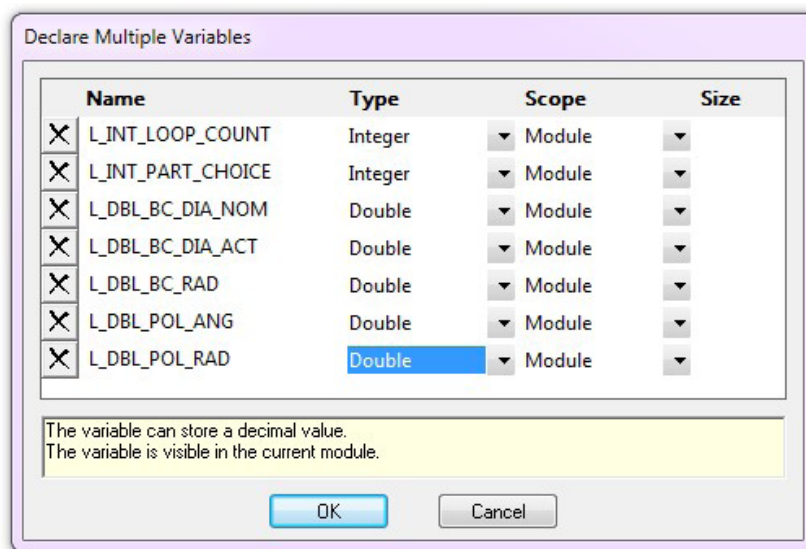
**L\_STR\_FEAT\_NAME\_NUM** - Concatenated feature and loop counter

**L\_STR\_ERROR\_MSG** - A text string to indicate if the hole is over or under specification

**L\_STR\_FILE\_PATH** - Holds the file path of the output file

**L\_STR\_FILE\_NAME** - Holds the name of the output file

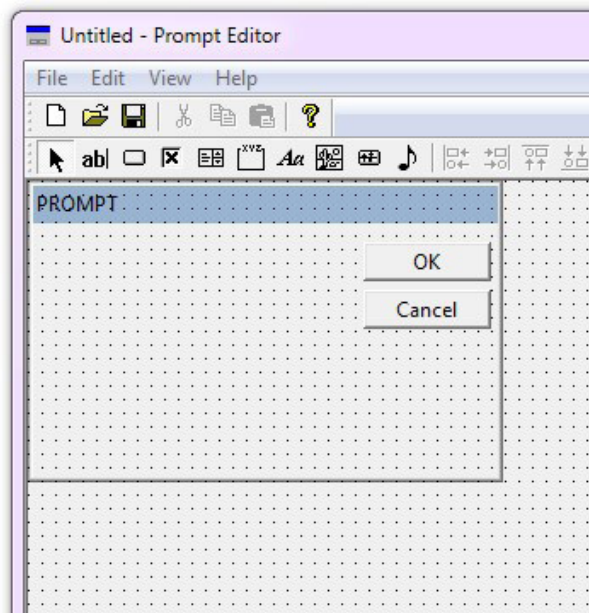
**L\_STR\_FILE\_PATH\_NAME** - Holds the concatenated path and name




**GUIDANCE NOTE:** There is a limit to how many variables that can be declared in 'Declare Multiple Variables', so it must be called twice to declare all required variables.

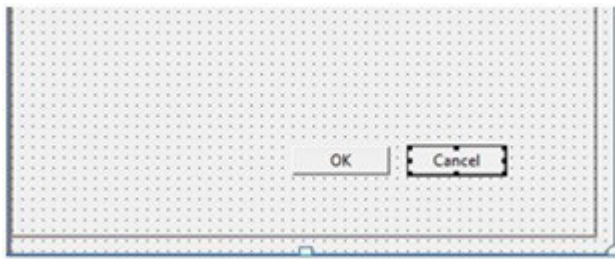
## 8 Create a prompt

From the 'High Level' menu, select 'Prompt' (not 'Simple Prompt'), click on 'Prompt' field, and edit the height and width in the properties area on the right side of the window. This will give room to add items to the prompt later. The figures shown in yellow for height and width are for indication purposes only. Size the box as required.

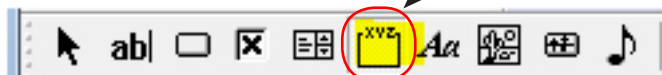


Properties	Templates
Dialog	
Caption	PROMPT
Height	250
Left	0
Top	0
Width	300
Variable	Var1
Declare variable	True
Template	

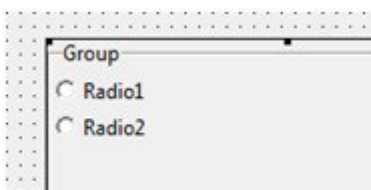
After clicking on the selection tool , objects in the PROMPT window can be moved and resized. For example, the 'OK' and 'Cancel' button can be moved to the bottom.



Select: 'Group' tool.



Drag a box to create a list of radio buttons:



Edit the 'Button Text' field with comma separated text to indicate the names required for the radio buttons. This indicates the choices that the operator can select (e.g. 'Small,Medium,Large '). A group tool allows the operator to choose only one choice at a time.

Edit the 'Text' field to a suitable name for the group field (e.g. 'Size Selector' ). This places a title at the top of the group field.

Edit the 'Variable' field with a suitable variable (e.g. L\_INT\_PART\_CHOICE). When a choice is made, the variable will be assigned an integer value, based on the position (e.g. Small=1, Medium=2, Large=3).

The figures shown for height, left, top and width are for indication purposes only. Size and position as required.

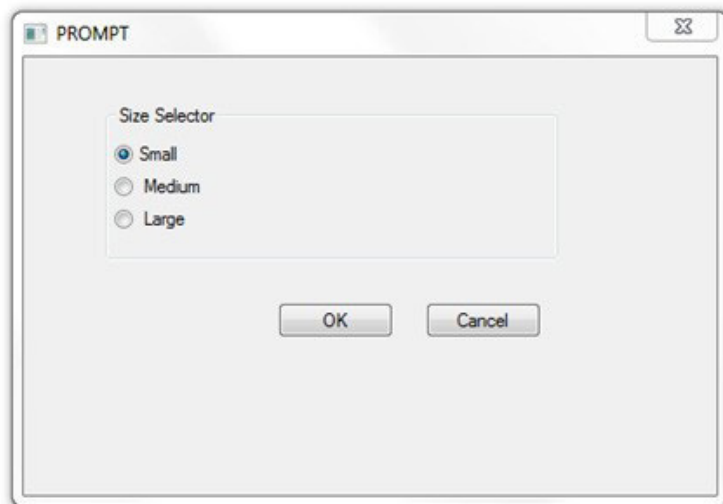
Group0	
Button Text	Small,Medium,Large
Height	70
Left	30
Name	Group0
Text	Size Selector
Top	30
Variable	L_INT_PART_CHOICE
Width	70

Click 'File' and choose 'Exit and update DMIS code' to save the prompt to the DMIS program.

**GUIDANCE NOTE:** It is not necessary to 'Save', 'Save As' or 'Save as Template', since the prompt is defined as DMIS code. However, a previously saved prompt can be opened later.

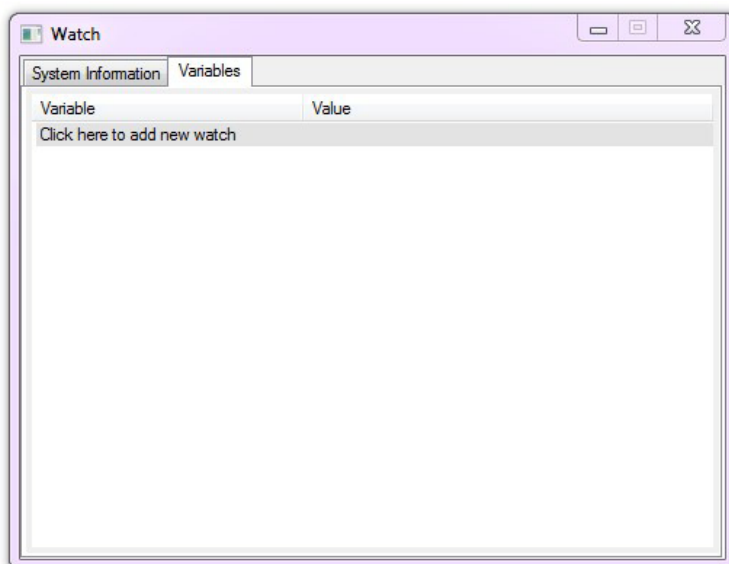
## 8.1 Resulting prompt

When the program runs the prompt code, the user can select from the choices in the group field (e.g. Small, Medium or Large). The advantage of the group tool is that the user can only choose from choices provided, eliminating the possibility that the user could miss-type the information. For the purpose of this tutorial select 'Small'.

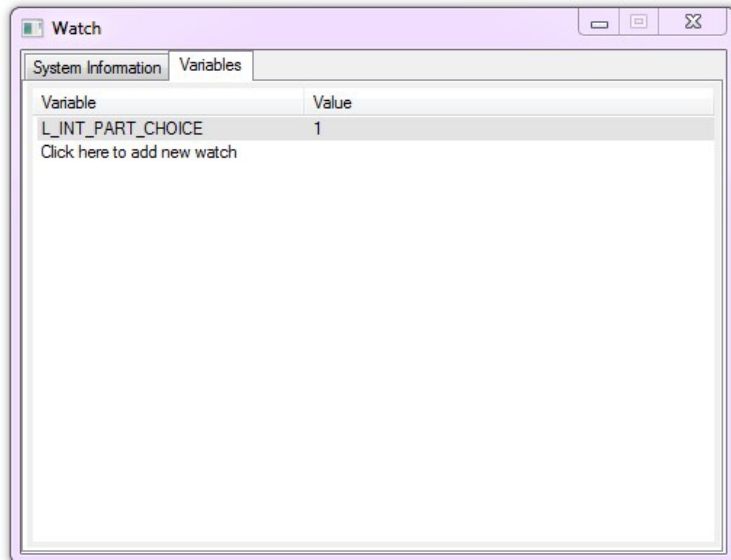


## 8.2 Variable watch window

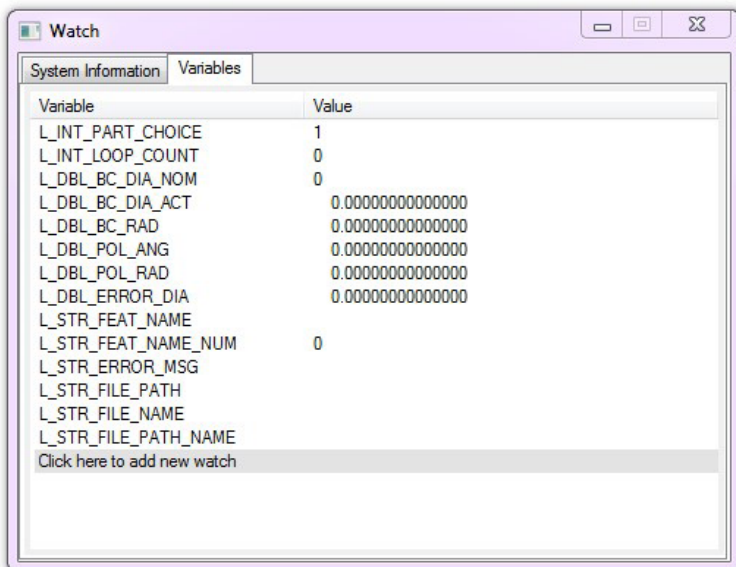
Having declared variables and started to use them, such as in this prompt, it is useful to open the variable 'Watch' window. This enables the programmer to confirm that the variable is functioning correctly. To start this go to 'High Level' then select 'Watch Variables' and the following window will appear.



Type the name of the variable(s) to watch and the value assigned to it / them will be displayed below the 'Value' label.

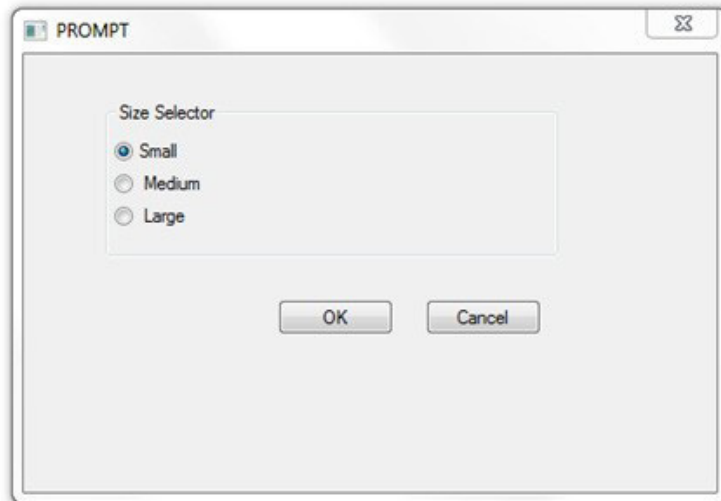


It is advisable when using variables to add them all to the Watch window so that they can be monitored at any given time .



### 8.3 Assign the PCD size using inputs from a prompt

When the user selects the size, the variable for that control is assigned automatically (e.g. choosing 'Small' assigns the value 1 to the variable L\_INT\_PART\_CHOICE).

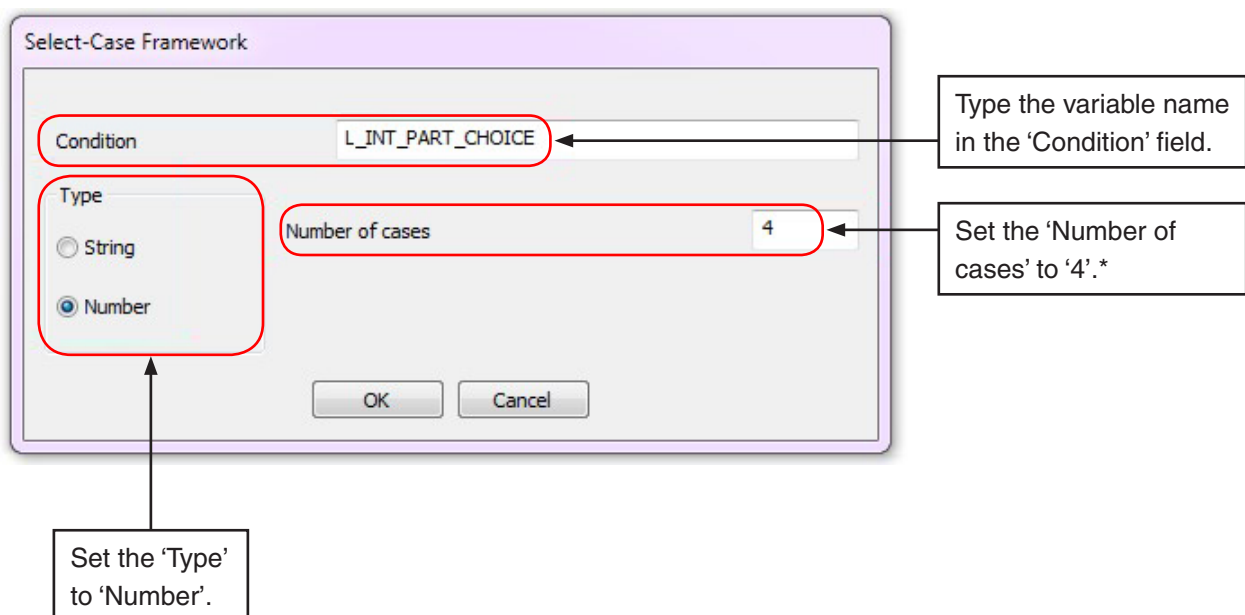


### 8.4 Select-Case

A 'Select-Case' controls the execution path, based on the value of a variable. This is more efficient than using a complicated set of 'IF-ELSE' statements.

This conditional structure is used to set the PCD diameter value based on the choice of the three radio buttons in the prompt.

From the 'High Level' menu, click on 'Select-Case', fill in details as below and click 'OK':



\* This figure should always be set to one more than required as the first is always set to 0.

**Sample DMIS code: (before assignment of values)**

```
SELECT/L_INT_PART_CHOICE
```

```
CASE/0
```

```
$$ Add your case-specific commands here
```

```
ENDCAS
```

```
CASE/1
```

```
$$ Add your case-specific commands here
```

```
ENDCAS
```

```
CASE/2
```

```
$$ Add your case-specific commands here
```

```
ENDCAS
```

```
CASE/3
```

```
$$ Add your case-specific commands here
```

```
ENDCAS
```

```
DFTCAS
```

```
$$ Add your default case commands here ENDCAS
```

```
ENDSEL
```

---

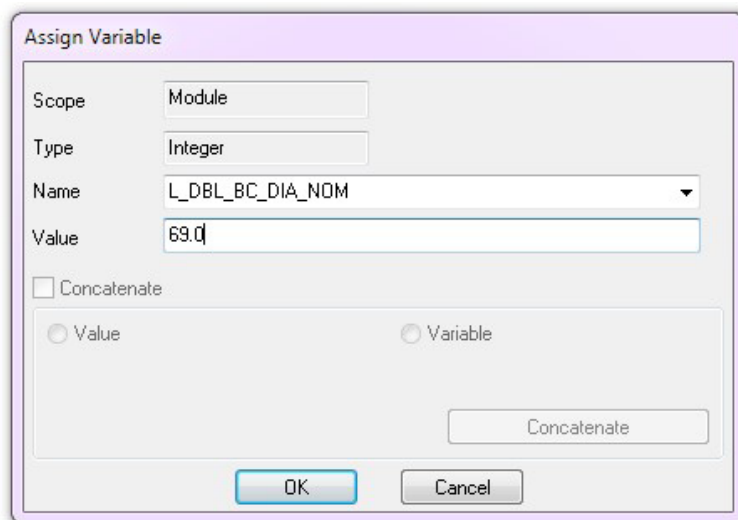
**GUIDANCE NOTE:** All cases from 0 to DFTCAS are added automatically.

---

## 8.5 Assign the values in the 'Select-Case'

Values are now assigned to the Bolt Circle variable.

Go to 'High Level' then select 'Assign' and set the variables.



The 'Assign Variable' dialog box is shown with the following settings:

- Scope: Module
- Type: Integer
- Name: L\_DBL\_BC\_DIA\_NOM
- Value: 69.0
- ☐ Concatenate
- ☐ Value
- ☐ Variable
- Concatenate button
- OK button
- Cancel button

There can be no case 0, (inputs should be 1,2 or 3), so remove the section (CASE/0 to ENDCAS) completely.

For case 1, assign a numerical value for a 'Small' hole diameter:-

E.g. `L_DBL_BC_DIA_ NOM=ASSIGN/69.0`

For case 2, assign a numerical value for a 'Medium' hole diameter:-

E.g. `L_DBL_BC_DIA_ NOM=ASSIGN/112.4`

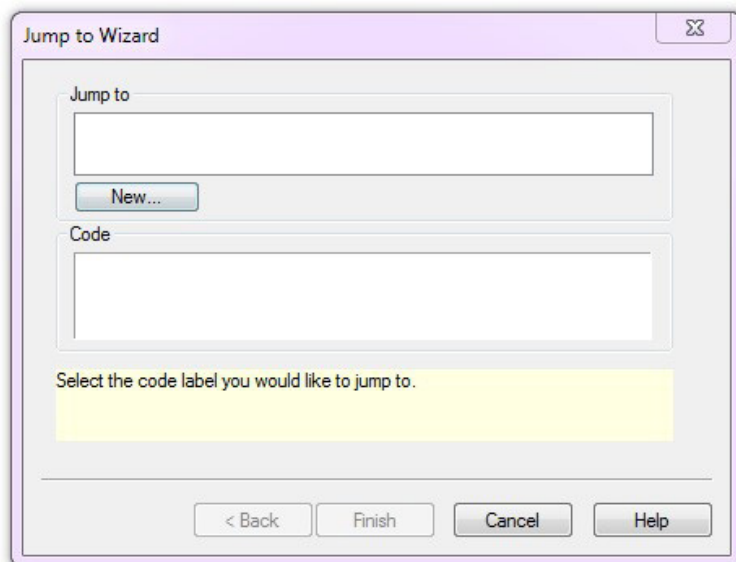
For case 3, assign a numerical value for a 'Large' hole diameter:-

E.g. `L_DBL_BC_DIA_ NOM=ASSIGN/151.5`

The numerical values used in cases 2 and 3 above are for reference only and do not apply to any features on the training block.

For case DFTCAS (i.e. anything other than 1,2 or 3), add a JUMPTO to a LABEL above the 'Prompt' line.

To add the 'JUMPTO', click 'High Level' then select 'Jump to Wizard'. Next click the 'New' button.



Scroll up to line above the previously declared PROMPT Var1 and type 'RETRY' into the Name field:-



Sample DMIS code (after assignment of values):

```
SELECT/L_INT_PART_CHOICE
```

```
CASE/1
```

```
L_DBL_BC_DIA_ NOM=ASSIGN/69.0
```

```
ENDCAS
```

```
CASE/2
```

```
L_DBL_BC_DIA_ NOM=ASSIGN/112.4
```

```
ENDCAS
```

```
CASE/3
```

```
L_DBL_BC_DIA_ NOM=ASSIGN/151.5
```

```
ENDCAS
```

```
DFTCAS
```

```
JUMPTO/(RETRY)
```

```
ENDCAS
```

```
ENDSEL
```

---

**GUIDANCE NOTE:** Adding a label is a good example of where adding text / program code manually maybe quicker than using the menu system. To add this command manually place the cursor at the required position in the program then use 'Control + I' to enter the required label enclosed by brackets i.e. (RETRY).

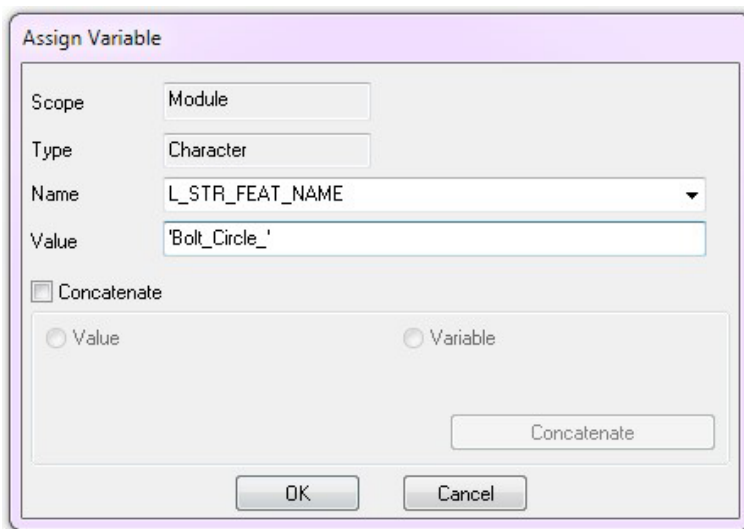
---

## 9 Assigning further variables

Variables are assigned in order to give them a value. There are various ways to accomplish this. Here are three examples:

1. Assign them directly with a value:- `L_INT_TEST=ASSIGN/123`
2. Use of a formula:- `L_INT_TEST=ASSIGN/L_INT_MULTIPLIER * 6`
3. Obtaining from a measured feature:- `L_DBL_ACT_DIA=OBTAIN/FA(CIR001),10`

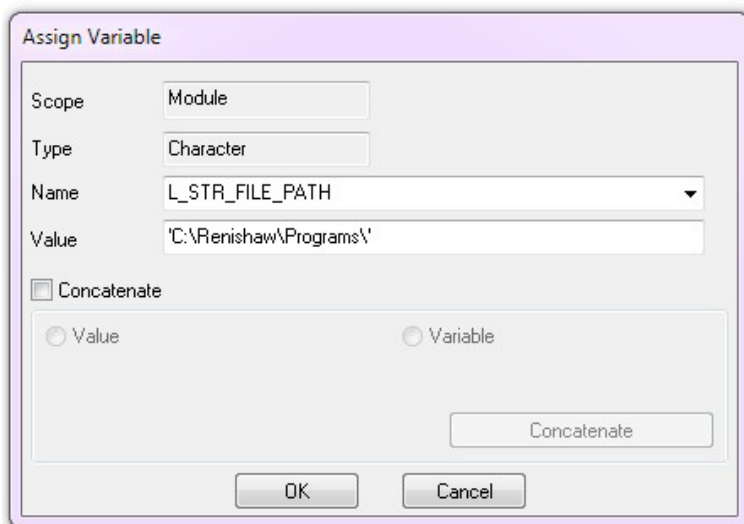
Go to 'High Level' then select 'Assign' and set the feature name variable. When assigning Character (String) variables it is essential to enclose the required text in single quotes (').



The 'Assign Variable' dialog box is shown with the following settings:

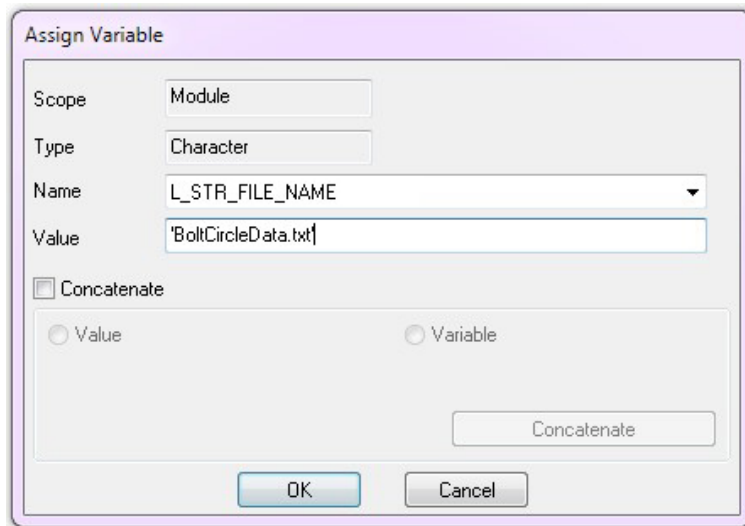
- Scope: Module
- Type: Character
- Name: L\_STR\_FEAT\_NAME
- Value: 'Bolt\_Circle\_'
- ☐ Concatenate
  - ☐ Value
  - ☐ Variable
- Buttons: OK, Cancel, Concatenate

Assign the value of the path where the data will go as well as the file name.



The 'Assign Variable' dialog box is shown with the following settings:

- Scope: Module
- Type: Character
- Name: L\_STR\_FILE\_PATH
- Value: 'C:\Renishaw\Programs\'
- ☐ Concatenate
  - ☐ Value
  - ☐ Variable
- Buttons: OK, Cancel, Concatenate



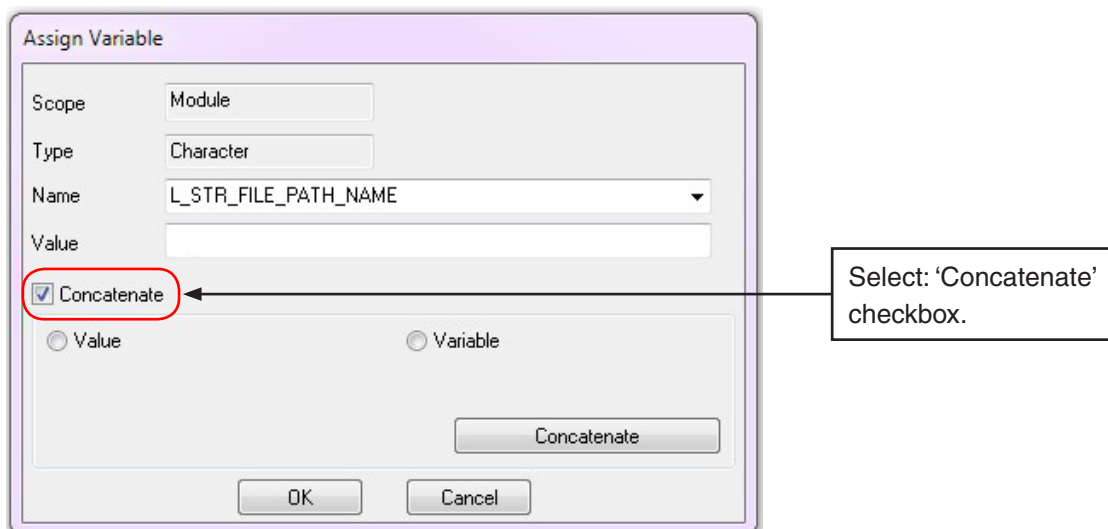
The 'Assign Variable' dialog box is shown with the following settings:

- Scope: Module
- Type: Character
- Name: L\_STR\_FILE\_NAME
- Value: 'BoltCircleData.txt'
- ☐ Concatenate
- ☐ Value
- ☐ Variable
- Buttons: OK, Cancel, Concatenate

## 9.1 Concatenation of a file name

Concatenation (combining) is used to add multiple pieces of information to a single variable.

The path and file name variables will be combined to illustrate this concept. Use the variable L\_STR\_FILE\_PATH\_NAME to hold the concatenated value. To enable the 'Concatenate' tool, select the 'Concatenate' check box (this function will only be available when you select a character variable from the 'Name' drop down list).

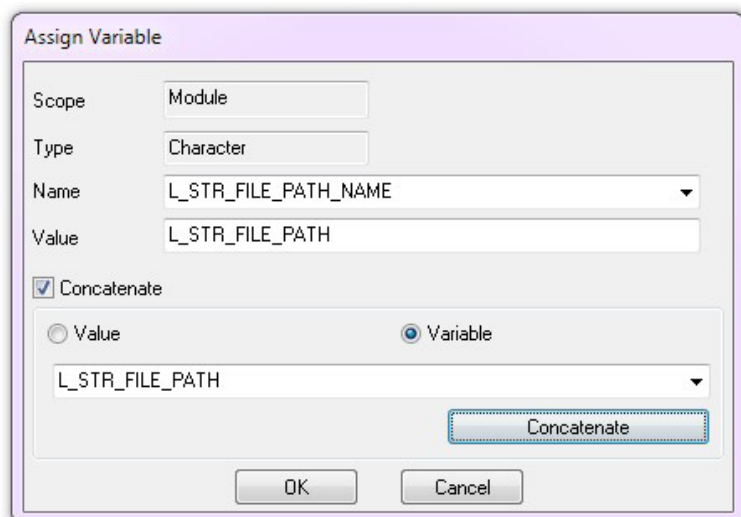


The 'Assign Variable' dialog box is shown with the following settings:

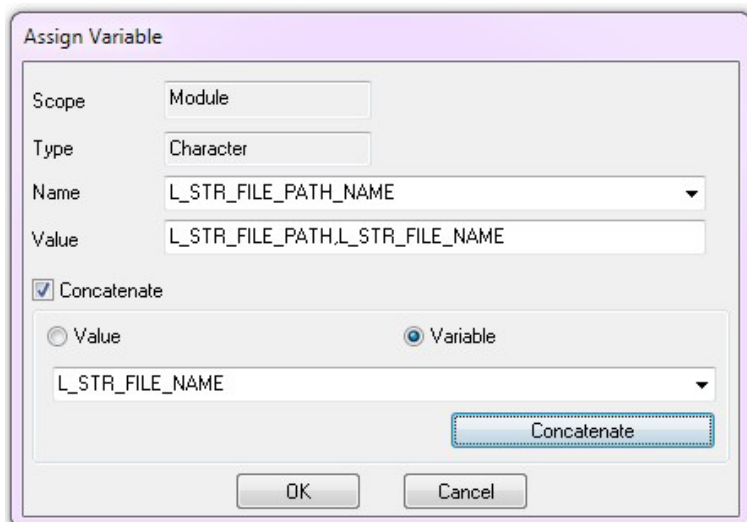
- Scope: Module
- Type: Character
- Name: L\_STR\_FILE\_PATH\_NAME
- Value: (empty)
- ☒ Concatenate
- ☐ Value
- ☐ Variable
- Buttons: OK, Cancel, Concatenate

A red box highlights the 'Concatenate' checkbox, and an arrow points to it from a text box that says: "Select: 'Concatenate' checkbox."

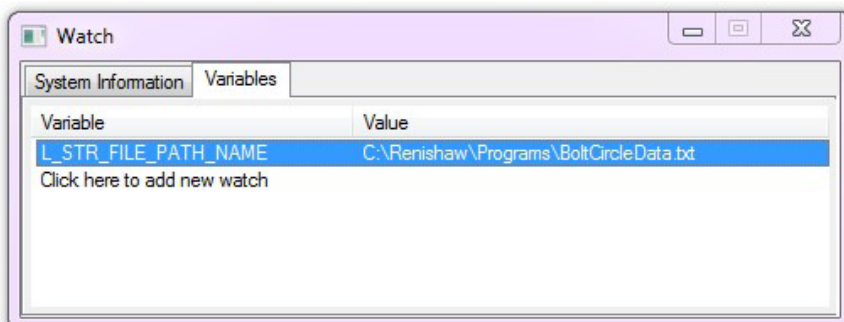
Select the 'Variable' radio button, browse for the variable 'L\_STR\_FILE\_PATH' in the drop down list then press the 'Concatenate' button.



Now browse for the variable 'L\_STR\_FILE\_NAME' in the drop down list then press the 'Concatenate' button, then click 'OK'.



In the 'Watch' variables window, the value of the variable 'L\_STR\_FILE\_PATH\_NAME' should now contain the values of the pre-assigned 'L\_STR\_FILE\_PATH' and 'L\_STR\_FILE\_NAME' variables combined together.

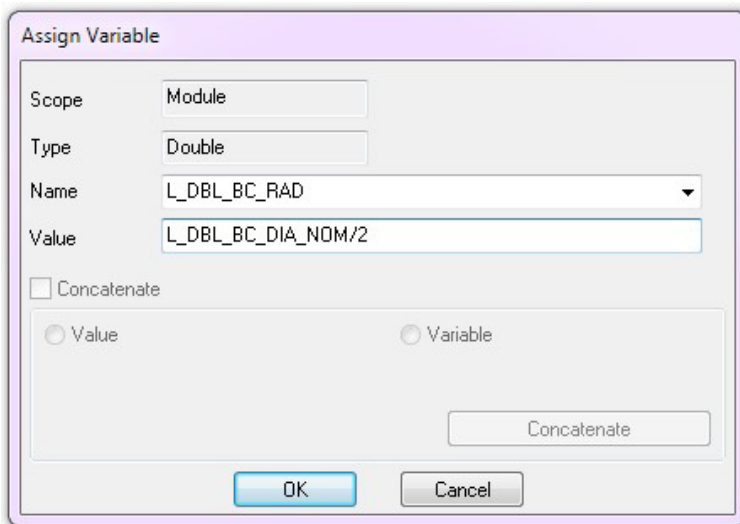


Sample Dmis code:-

```
L_STR_FILE_PATH_NAME=ASSIGN/CONCAT(L_STR_FILE_PATH,L_STR_FILE_NAME)
```

## 9.2 Assigning a value with a calculation

Go to 'High Level' then select 'Assign' and add a formula to divide the pitch circle diameter by two to calculate the radial distance.

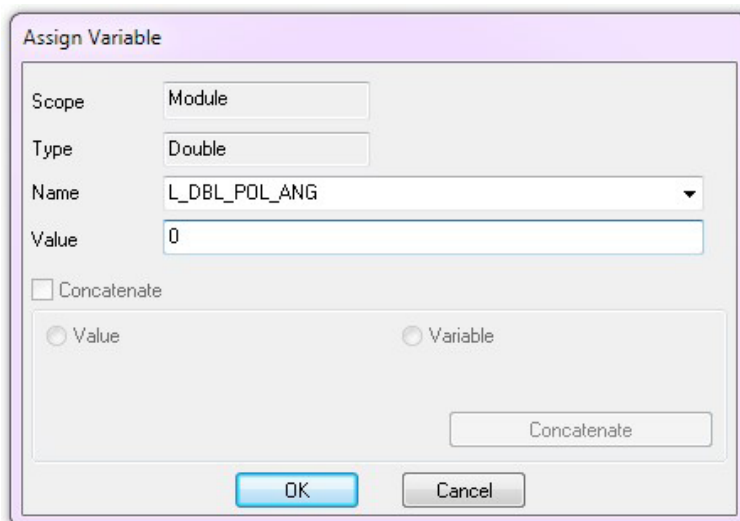


The 'Assign Variable' dialog box is shown with the following settings:

- Scope: Module
- Type: Double
- Name: L\_DBL\_BC\_RAD
- Value: L\_DBL\_BC\_DIA\_NOM/2
- ☐ Concatenate
- ☐ Value
- ☐ Variable
- Concatenate button
- OK button
- Cancel button

## 9.3 Assign the hole variables

Set the value of the polar angle variable to zero.

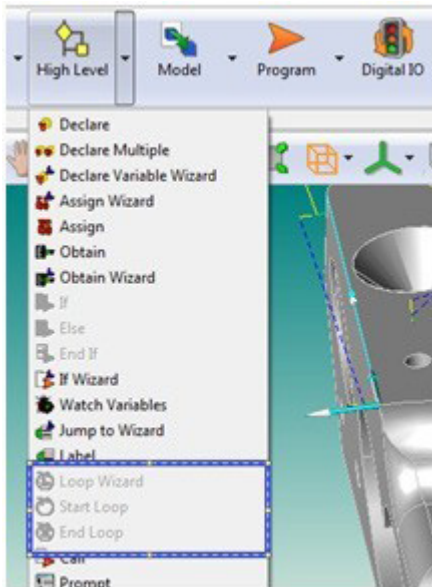


The 'Assign Variable' dialog box is shown with the following settings:

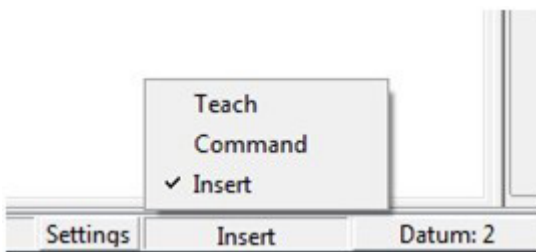
- Scope: Module
- Type: Double
- Name: L\_DBL\_POL\_ANG
- Value: 0
- ☐ Concatenate
- ☐ Value
- ☐ Variable
- Concatenate button
- OK button
- Cancel button

## 10 Creating a loop

A loop runs the same code a given number of times. Each time the program runs through the loop is called an 'Iteration'. This makes it possible to reduce the amount of code needed during a repetitive measurement.



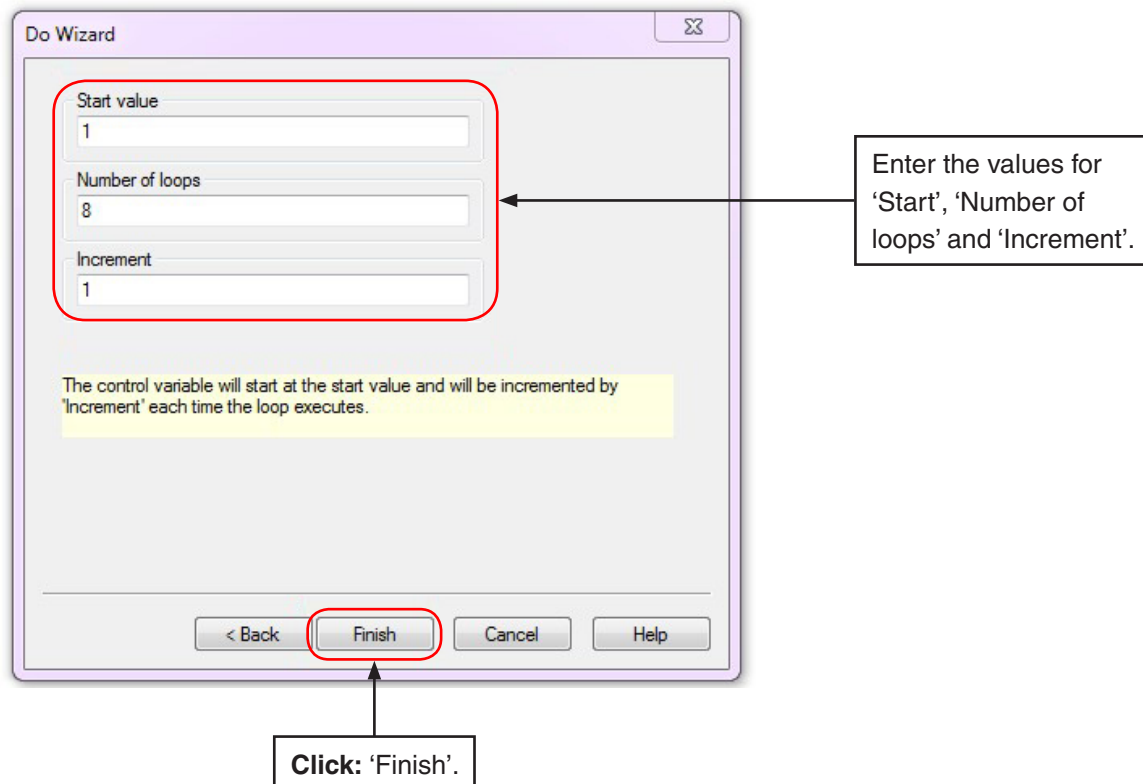
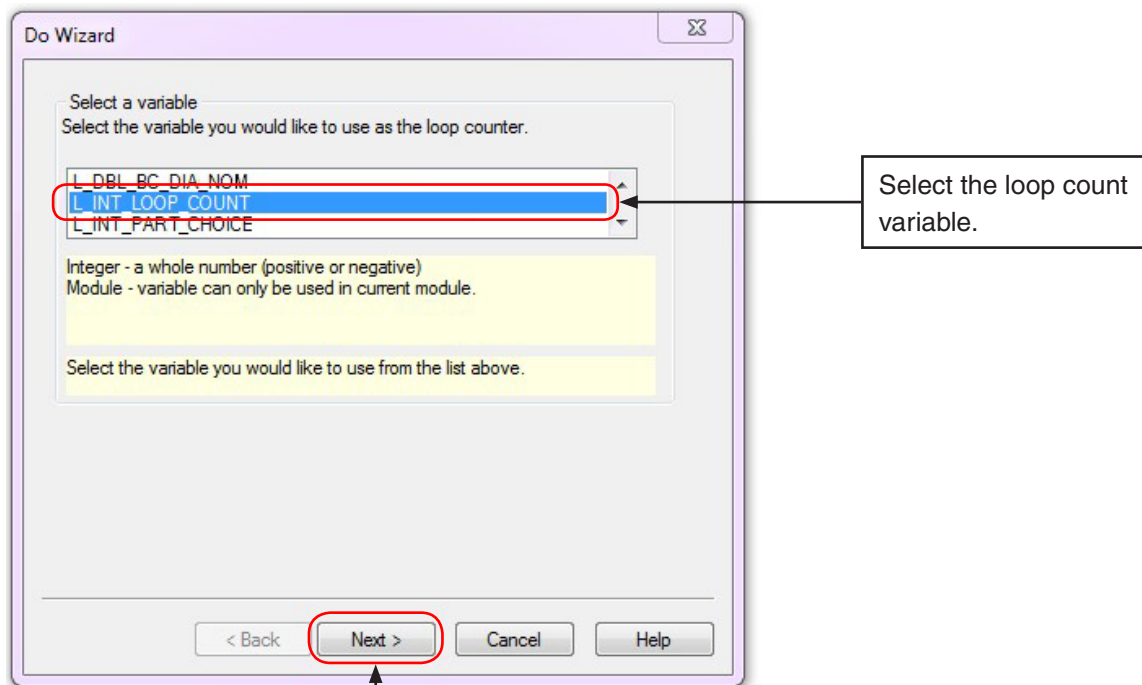
Click 'High Level', note that the loop buttons are not functional if MODUS is in 'Teach' or 'Command' mode.



Go to the bottom tool strip and change the selection from 'Teach' to 'Insert'. This will allow the loop commands to be selected.

Refer to the "Teach, Command and Insert Modes" help file for further explanation.

Select 'High Level' and 'Loop' wizard:



Once the commands have been inserted into the program ensure you change back to 'Teach' mode at the earliest opportunity.

### Sample DMIS code

The following code is added to the program:

```
DO/L_INT_LOOP_COUNT,1,8,1
```

```
$$Enter loop code here
```

```
ENDDO
```

---

**GUIDANCE NOTE:** The comment 'Enter loop code here' can be deleted. It indicates where to place the code needed to run inside the loop.

---

## 10.1 Add the hole measurement into the loop

The first hole of the bolt pattern will be measured on the right (+X).

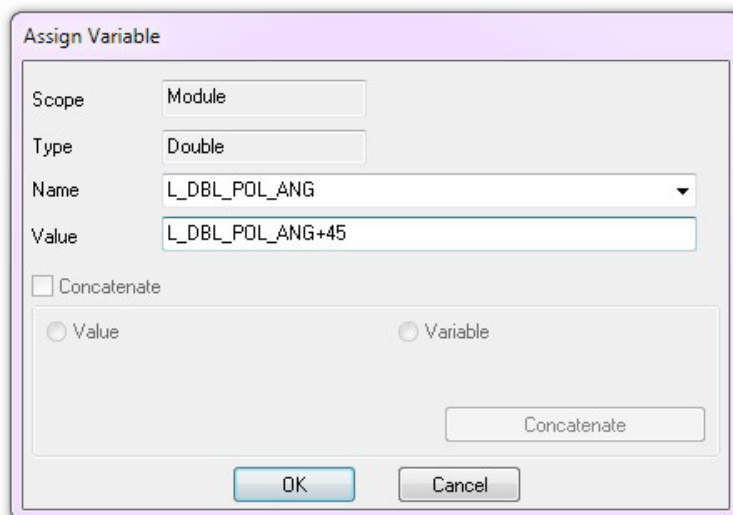
Set MODUS to measure in polar co-ordinates then measure a hole in AUTOMODE on the right, or plus X side of the hole pattern.

---

**GUIDANCE NOTE:** The use of AUTOMODE in this tutorial is for simplification purposes.

---

After the ENDMES of the hole, add an assignment statement to add 45 to the polar angle variable as shown below. This will cause the next hole to be measured in the next iteration of the loop.



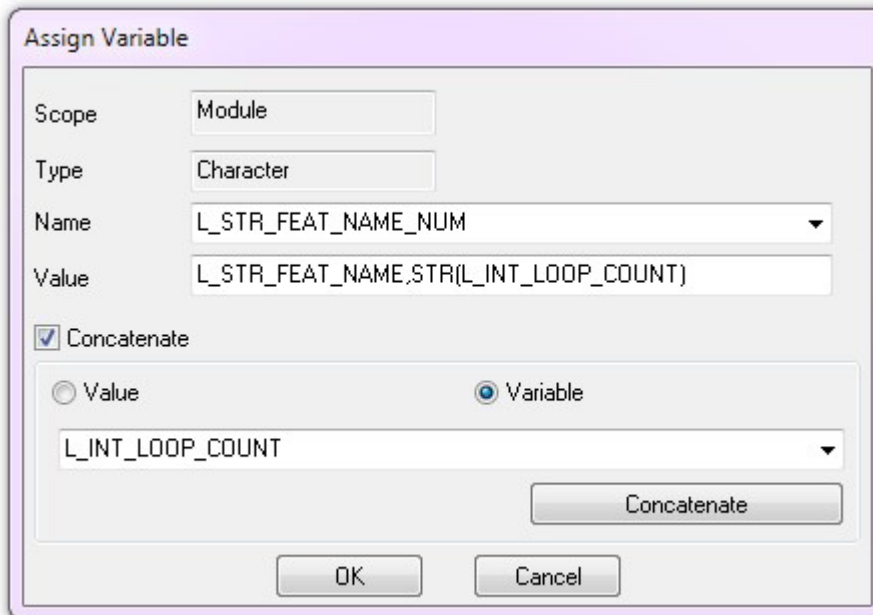
The image shows a software dialog box titled "Assign Variable". It contains the following fields and controls:

- Scope:** A text box containing "Module".
- Type:** A text box containing "Double".
- Name:** A dropdown menu showing "L\_DBL\_POL\_ANG".
- Value:** A text box containing "L\_DBL\_POL\_ANG+45".
- Concatenate:** An unchecked checkbox.
- Radio Buttons:** Two radio buttons labeled "Value" and "Variable", both of which are unselected.
- Buttons:** At the bottom are "OK" and "Cancel" buttons. A "Concatenate" button is located to the right of the radio buttons.

## 10.2 Edit the hole nominal

Use the concatenation tool in the Assign Variable dialogue to combine the feature name and the loop counter.

This 'Assign/Concat' function should be placed between the 'Do' line and the feature measurement block.

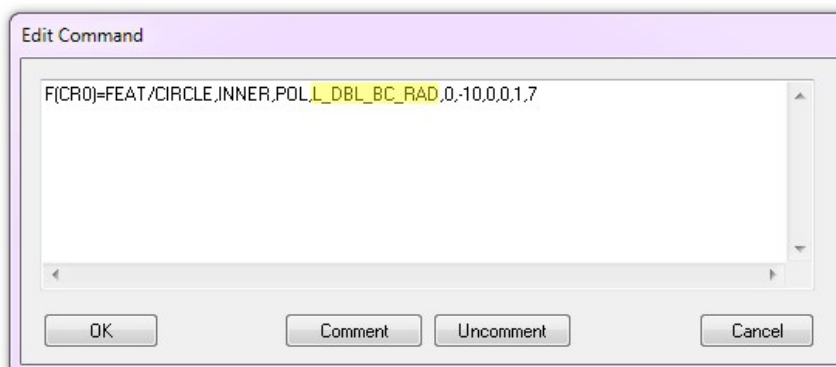


The following three 'Edit' functions should be carried out at the same time.

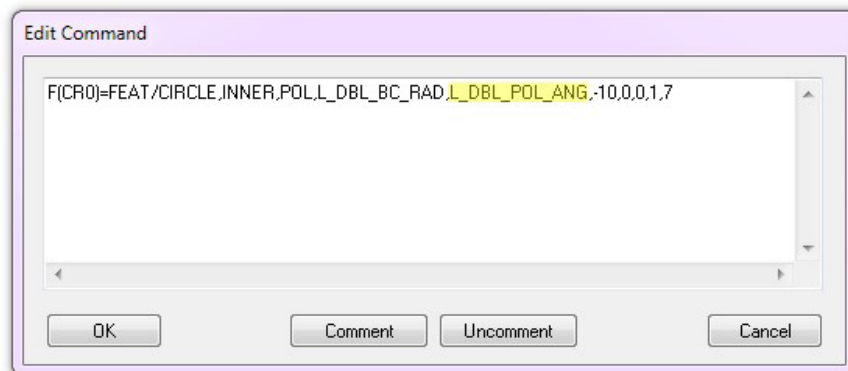
Replace the polar radius value with the corresponding variable. This must be done by text editing. Therefore, do not double click on the line. Use 'CTRL + E' to text edit the feature.

Code to be edited:

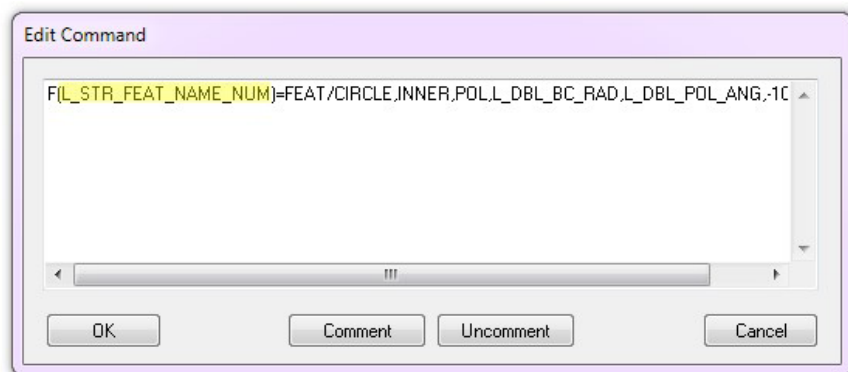
```
F(CRO)=FEAT/CIRCLE,INNER,POL,34.5,0,-10,0,0,1,7
```



Replace the polar angle value with the corresponding variable by text editing.



Replace the feature name with the corresponding variable by text editing.



Do the same in the MEAS/CIRCLE command:



### Sample DMIS code - after editing

The following code is the loop with the hole measurement:

\$\$ Start of PCD loop

DO/L\_INT\_LOOP\_COUNT,1,8,1

L\_STR\_FEAT\_NAME\_NUM=ASSIGN/CONCAT(L\_STR\_FEAT\_NAME,STR(L\_INT\_LOOP\_COUNT))

MODE/AUTO,PROG,MAN

F(L\_STR\_FEAT\_NAME\_NUM)=FEAT/CIRCLE,INNER,POL,L\_DBL\_BC\_RAD,L\_DBL\_POL\_ANG,-  
10,0,0,1,7

MEAS/CIRCLE,F(L\_STR\_FEAT\_NAME\_NUM),7

ENDMES

L\_DBL\_POL\_ANG=ASSIGN/L\_DBL\_POL\_ANG+ 45

ENDDO

## 10.3 Construct the PCD

Once the loop has been run eight times, the PCD feature can be created using a best fit circle construction.

## 10.4 Alternative loop method

Up to now, this tutorial has determined the position of each hole using a variable polar position (e.g. L\_DBL\_BC\_ANG). An alternative method would be to use a constant Cartesian position & add a datum rotation at the end of the looped code. However, if this is the case, the datum label must also be variable. This is to overcome the fact that once the loop is complete, every individual feature will have the same nominal value and the same datum reference (e.g. D(1)). Each feature needs to be independently defined in the database in conformance of the DMIS standard v5.2.

### Sample DMIS code

The following code is the loop with the hole measurement:

\$\$ Start of PCD loop

DO/L\_INT\_LOOP\_COUNT,1,8,1

L\_STR\_FEAT\_NAME\_NUM=ASSIGN/CONCAT(L\_STR\_FEAT\_NAME,STR(L\_INT\_LOOP\_COUNT))

L\_STR\_DAT\_NAM=ASSIGN/CONCAT('HOLE\_PCS\_',STR(L\_INT\_LOOP\_COUNT))

F(L\_STR\_FEAT\_NAME\_NUM)=FEAT/CIRCLE,INNER,CART, L\_DBL\_BC\_RAD,0,-10,0,0,1,7

MEAS/CIRCLE,F(L\_STR\_FEAT\_NAME\_NUM),4

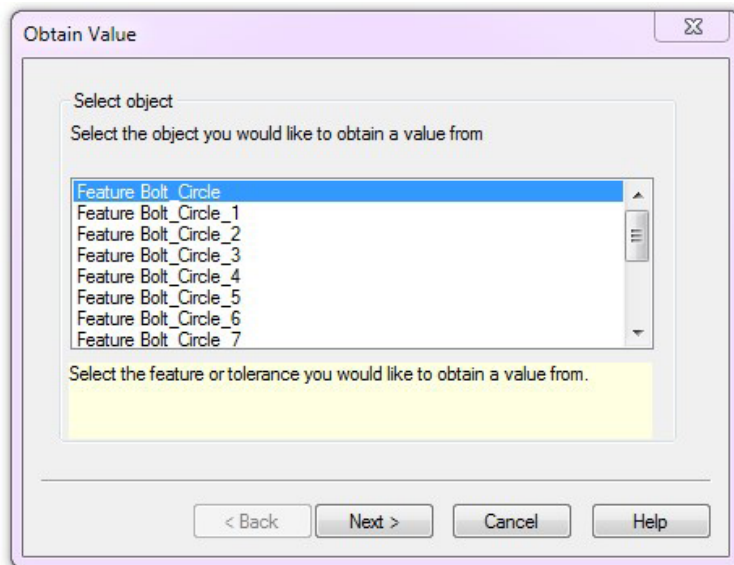
ENDMES

D(L\_STR\_DAT\_NAM)=ROTATE/ZAXIS,45

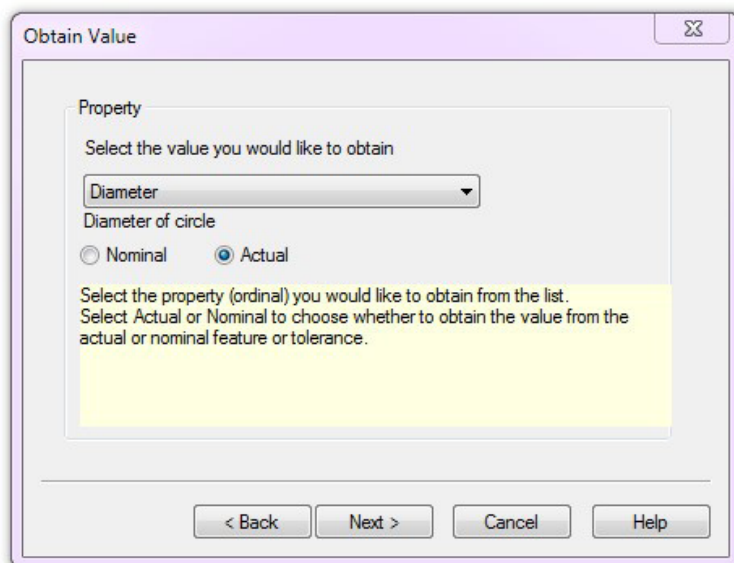
ENDDO

## 11 Obtaining a feature value

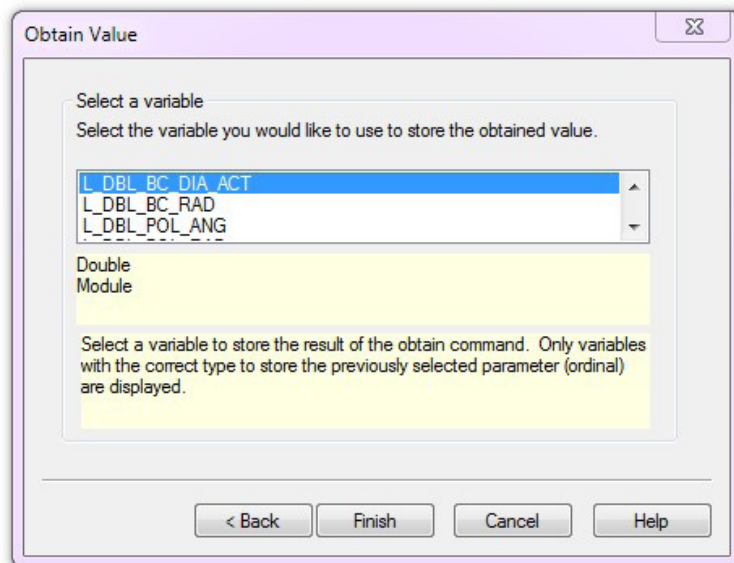
Now obtain the actual diameter and test for the error. Select 'Obtain Wizard' from the 'High Level' menu and choose the constructed PCD circle (eg. 'Bolt\_Circle').



Select the 'Actual' radio button then from the drop down menu click on 'Diameter'



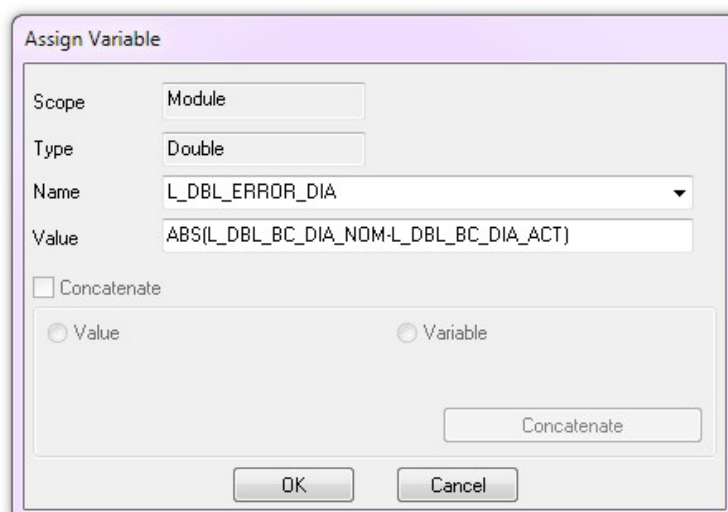
Select the relevant variable to hold the value obtained from the actual diameter and click 'Finish'.



Now the diameter error needs to be calculated.

In the 'Assign Variable' dialogue, select the variable to which the diameter error will be assigned from the 'Name' dropdown menu.

In the box 'Value', type the following text to subtract the variable containing the actual circle diameter from the variable containing the nominal diameter.



The use of the intrinsic function 'Absolute (ABS)' ensures the result will always be stated as a positive figure. The reason for this will become evident later in this tutorial.

---

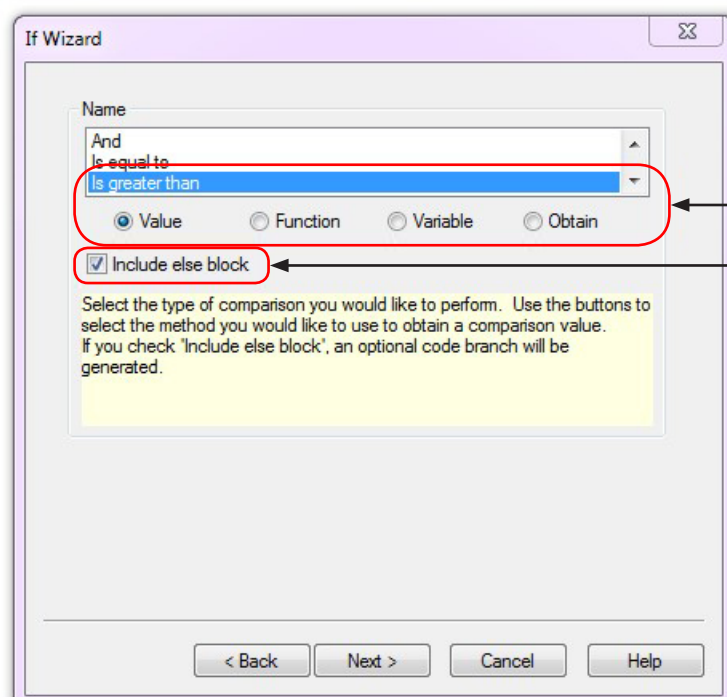
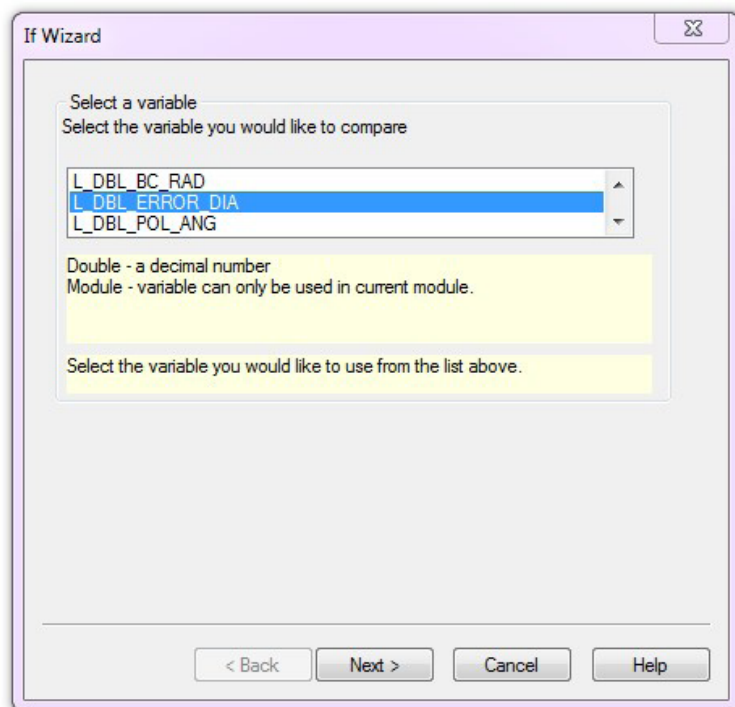
**GUIDANCE NOTE:** For further information on intrinsic functions see the DMIS standard.

---

## 11.1 Creating an IF statement

Now check the value of the error using an IF statement.

Click on 'High Level' then select 'If Wizard' from the drop down menu. Select the variable containing the diameter error.

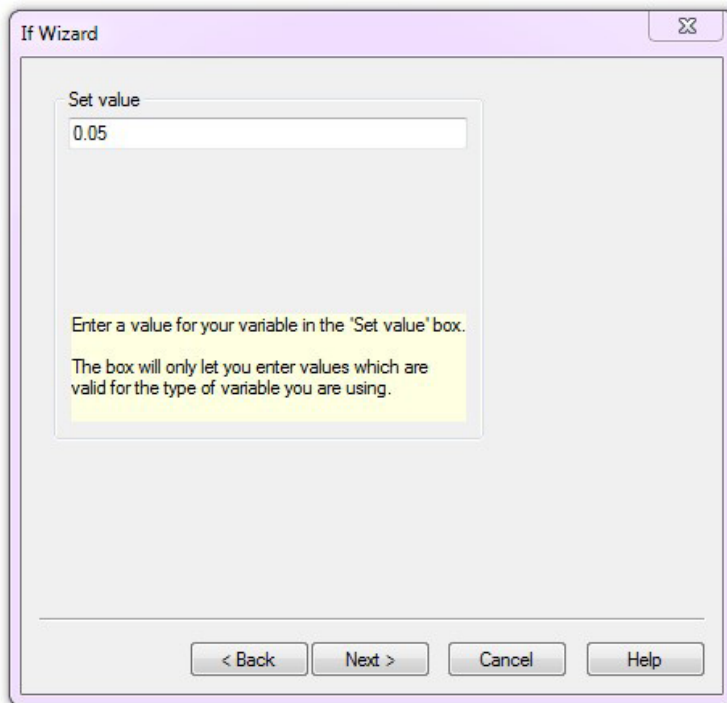


Select: 'Is greater than'.

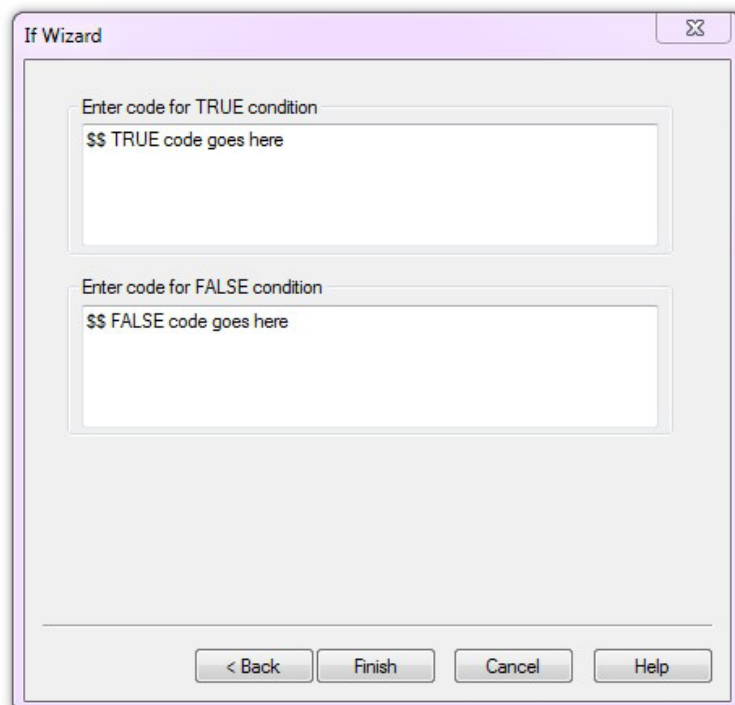
Click: 'Value'.

Click: 'Include else block'.

Set the value to a suitable tolerance value. This determines whether to execute the IF block or the ELSE block.



The 'If Wizard' dialog box is shown with a purple title bar. It contains a 'Set value' section with a text input field containing '0.05'. Below the input field, there is a yellow highlighted box with the text: 'Enter a value for your variable in the 'Set value' box. The box will only let you enter values which are valid for the type of variable you are using.' At the bottom, there are four buttons: '< Back', 'Next >', 'Cancel', and 'Help'.



The 'If Wizard' dialog box is shown with a purple title bar. It contains two text input fields. The first field is labeled 'Enter code for TRUE condition' and contains the text '\$\$ TRUE code goes here'. The second field is labeled 'Enter code for FALSE condition' and contains the text '\$\$ FALSE code goes here'. At the bottom, there are four buttons: '< Back', 'Finish', 'Cancel', and 'Help'.

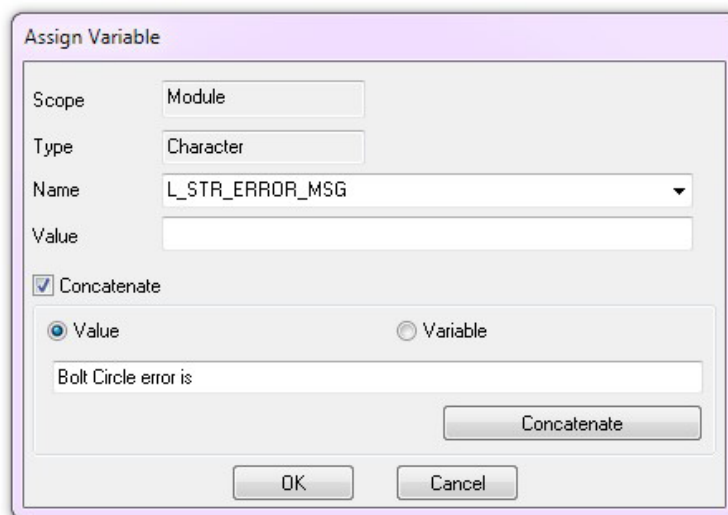
On completion of the above IF statement a TEXT / OPER should be added after the ENDIF statement as shown below. A blank TEXT / OPER line will have to be added to the program then edited to include the variable to achieve what is shown below.

The following sample code is generated:

```
IF/(L_DBL_ERROR_DIA.GT.0.050000)
$$ TRUE code goes here
ELSE
$$ FALSE code goes here
ENDIF
```

TEXT/OPER,L\_STR\_ERROR\_MSG

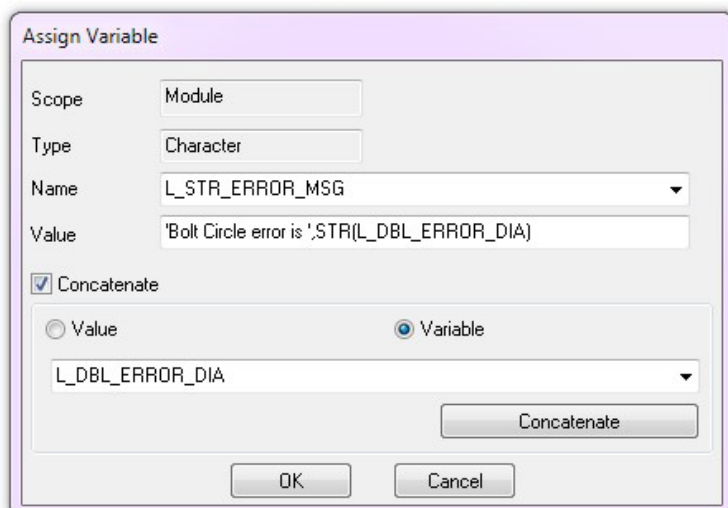
Add an assignment statement to combine text and the error value to the IF-TRUE section of the IF-ELSE statement. Using the concatenation tool in the Assign Variable dialogue, select the 'Value' radio button and type in the required hard-coded text (please note when using the concatenation tool there is no need to use inverted commas to denote text).



The 'Assign Variable' dialog box is shown with the following settings:

- Scope: Module
- Type: Character
- Name: L\_STR\_ERROR\_MSG
- Value: (empty text box)
- ☒ Concatenate
- ☒ Value
- ☐ Variable
- Text input: Bolt Circle error is
- Buttons: OK, Cancel, Concatenate

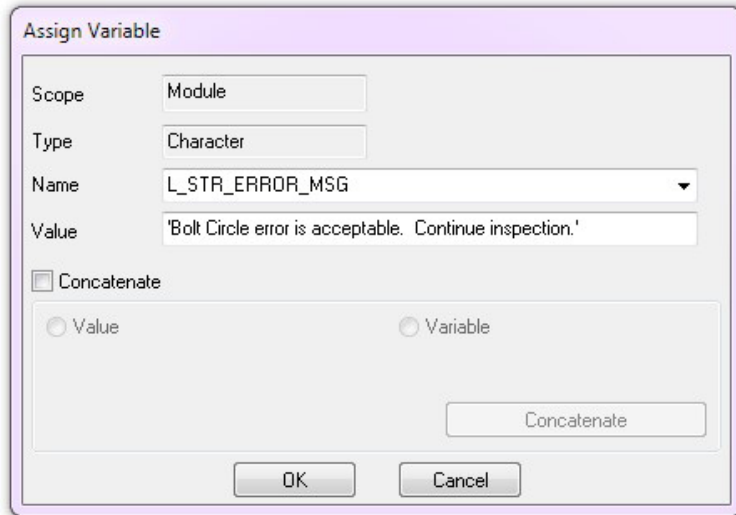
Now select the 'Variable' radio button, select the required variable from the pulldown menu and click 'Concatenate' then 'OK'.



The 'Assign Variable' dialog box is shown with the following settings:

- Scope: Module
- Type: Character
- Name: L\_STR\_ERROR\_MSG
- Value: 'Bolt Circle error is ',STR(L\_DBL\_ERROR\_DIA)
- ☒ Concatenate
- ☐ Value
- ☒ Variable
- Variable pulldown menu: L\_DBL\_ERROR\_DIA
- Buttons: OK, Cancel, Concatenate

Assign text for the ELSE section of the IF-FALSE statement indicating it is OK to continue. In this instance we are not using the concatenation tool therefore inverted commas must be used to denote text.



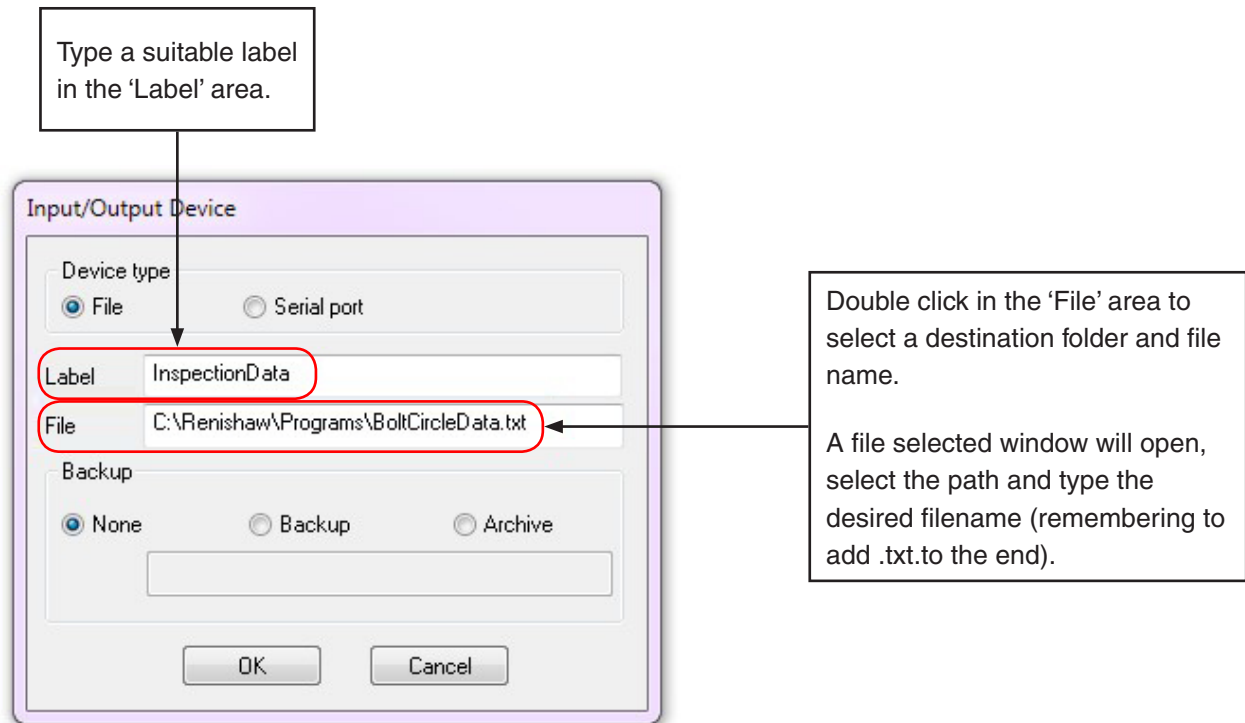
The image shows a dialog box titled "Assign Variable". It contains the following fields and controls:

- Scope:** A text box containing "Module".
- Type:** A text box containing "Character".
- Name:** A dropdown menu showing "L\_STR\_ERROR\_MSG".
- Value:** A text box containing "'Bolt Circle error is acceptable. Continue inspection.'".
- Concatenate:** A checkbox that is currently unchecked.
- Concatenation Options:** Two radio buttons, "Value" and "Variable", both of which are unselected.
- Concatenate Button:** A button labeled "Concatenate" that is disabled.
- OK and Cancel Buttons:** Two buttons at the bottom, "OK" and "Cancel", both of which are enabled.

## 12 Device files

A device file is an external file which MODUS can use for multiple purposes. In this case, the diameter error result, along with the message from the IF-ELSE block will be written to a text file.

Click 'High Level' and then select 'Device':

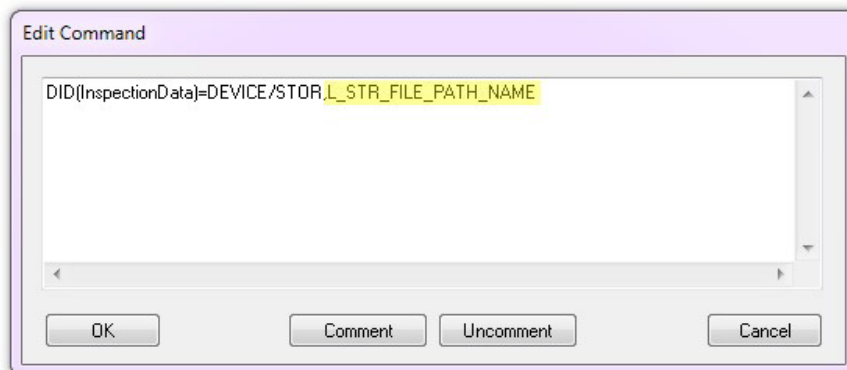


Right click on the line created and select Edit:

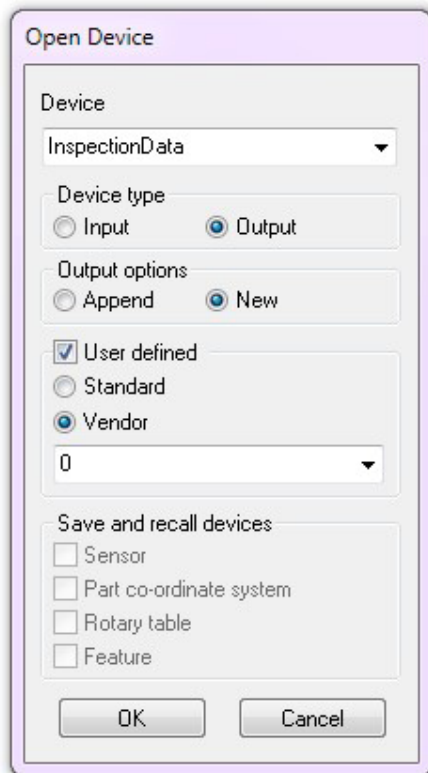
DID(InspectionData)=DEVICE/STOR,'C:\Renishaw\Programs\BoltCircleData.txt'

An Edit Command window will allow the text to be edited rather than opening the dialogue.

Replace the path and file name entered with the L\_STR\_FILE\_PATH\_NAME variable.



Click on 'High Level' then select 'Open':



Now open the device, select the following:

- Device name (from any declared devices)
- Output - this writes, rather than reads
- New - this writes a new entry rather than appending to an old entry
- User defined - Select 'Vendor' output format

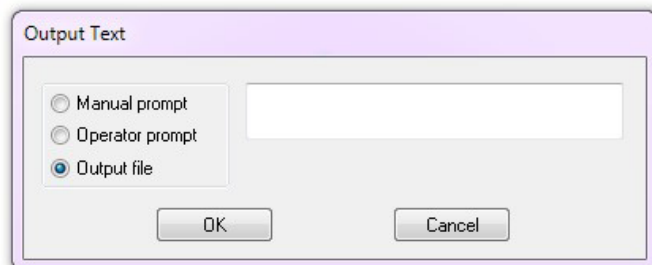
---

**GUIDANCE NOTE:** The following line must be in the program to define a vendor output format, which defines what is displayed in the relevant file, e.g. V(0)=VFORM/ALL (normally defined within a standard MODUS template)

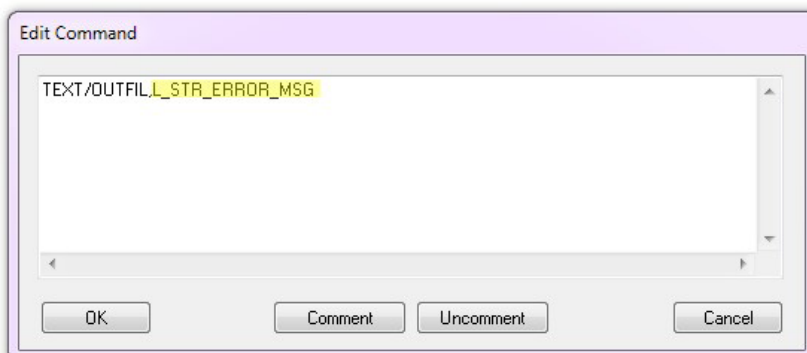
---

Any tolerance or text data that is output will now be sent to the file specified in the 'Device' command.

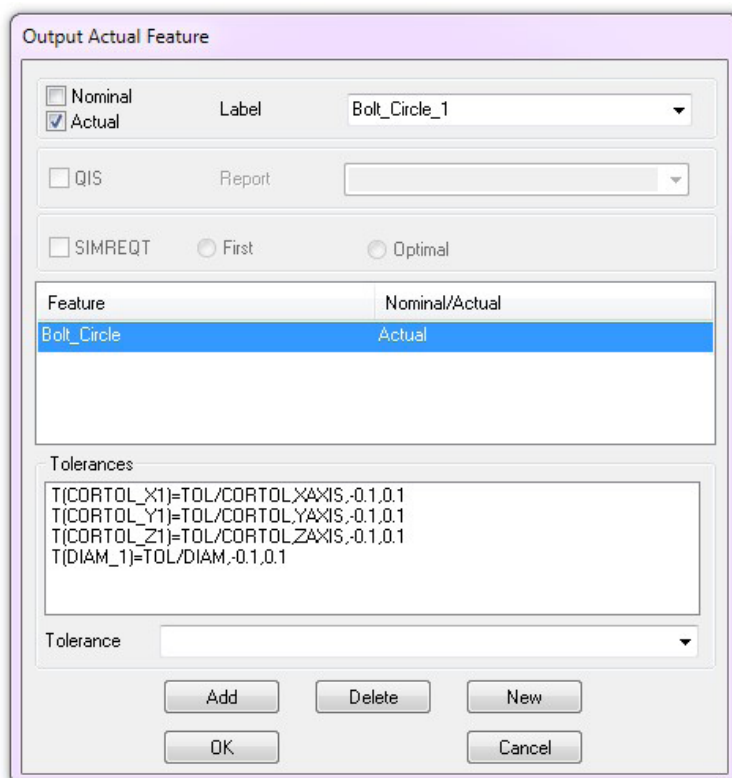
Click on 'Output' then select 'Text'; select 'Output file' radio button do not add any text (a variable will be used instead), click 'OK':



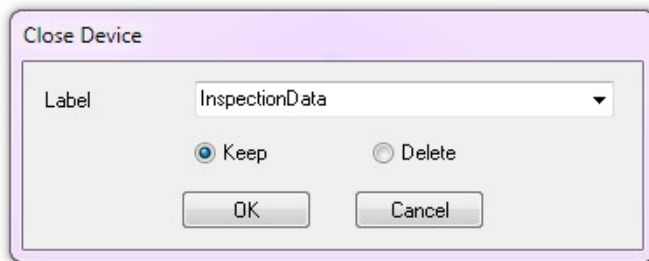
Edit the output text command, delete the single quotes and replace with the variable holding the error message.



Output the PCD and add tolerance values.



Close the device.



The data will appear in the file that was specified in the device command.

Find the location that was specified earlier and open the file to verify that the data is as expected.

PCD error is acceptable. Continue inspection

Circle:Bolt\_Circle

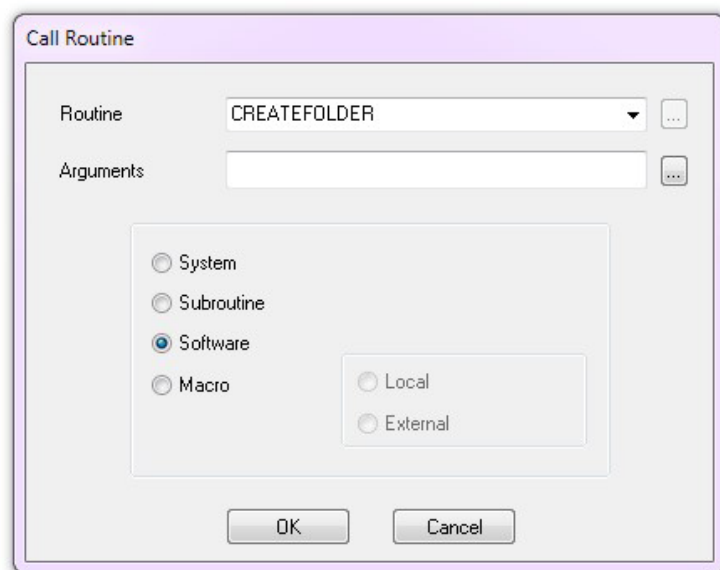
X-axis	0.001	0.000	-0.100	+0.100	0.001 ---*---
Y-axis	-0.000	0.000	-0.100	+0.100	-0.000 ---*---
Z-axis	-10.000	-10.000	-0.100	+0.100	0.000 ---*---
Diameter	69.00	69.00	-0.100	+0.100	0.000 ---*---

## 13 Creating folders and copying files

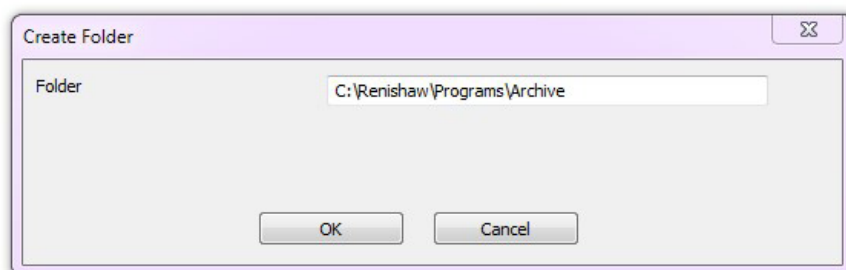
### 13.1 Creating folders

The file can now be copied to another location using a line of code in the DMIS program. This might be done to archive the results to a network location.

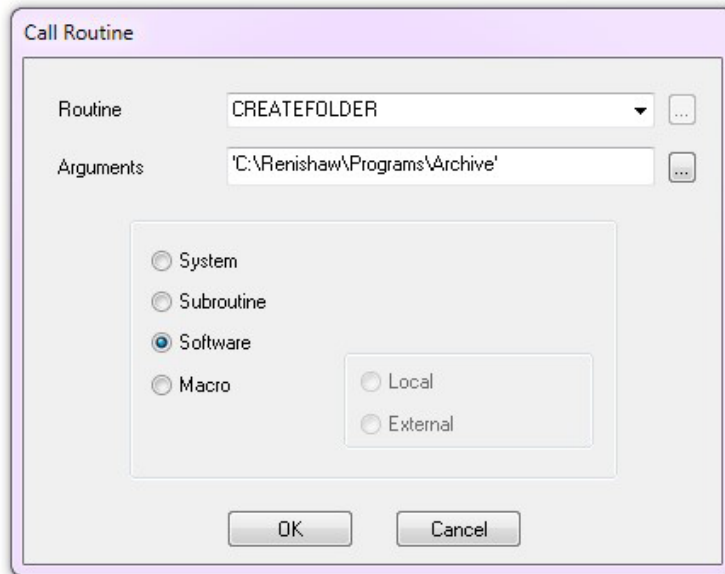
Click 'High Level' then select 'Call'. Click the 'Software' radio button and then select 'CREATEFOLDER' from the 'Routine' drop down menu



Now click on the 'Arguments' browse button and type the complete path in the prompt window (without quotation marks) then click 'OK'.



The path is placed in the Arguments text box.

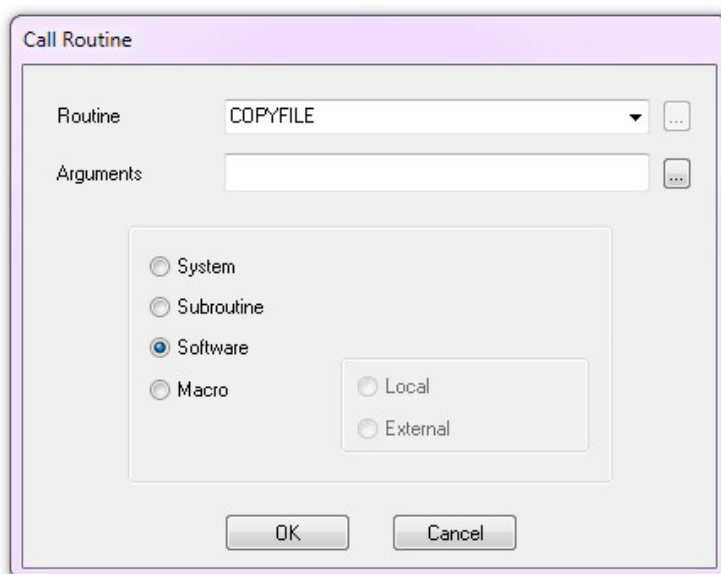


Click 'OK' to complete the process. This will create a new folder at the desired location. If a folder of this name already exists in this location the command is ignored.

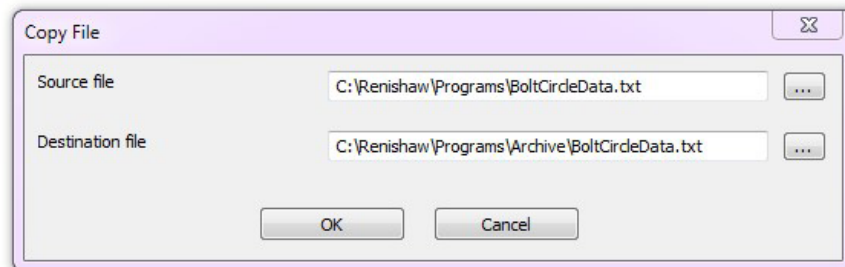
## 13.2 Copy file

Files can be copied using a similar technique.

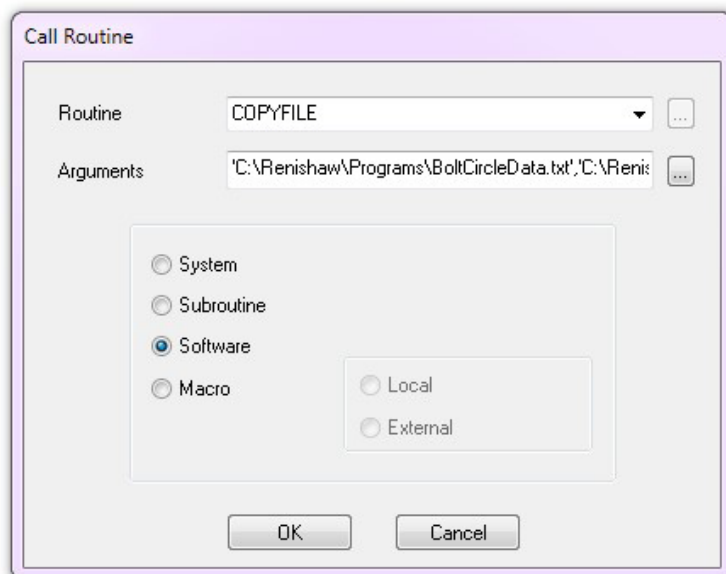
Click 'High Level' then select 'Call'. Click the 'Software' radio button and then select 'COPYFILE' from the 'Routine' drop down menu & click on the 'Arguments' browse button.



Click on the 'Source file' browse button and search for the file to copy or, alternatively, type the complete path in the prompt window (without quotation marks). Now click on the 'Destination file' browse button and search for the directory where the file is to be copied (adding the file name to this destination). Click 'OK'.



The paths are placed in the Arguments text box.

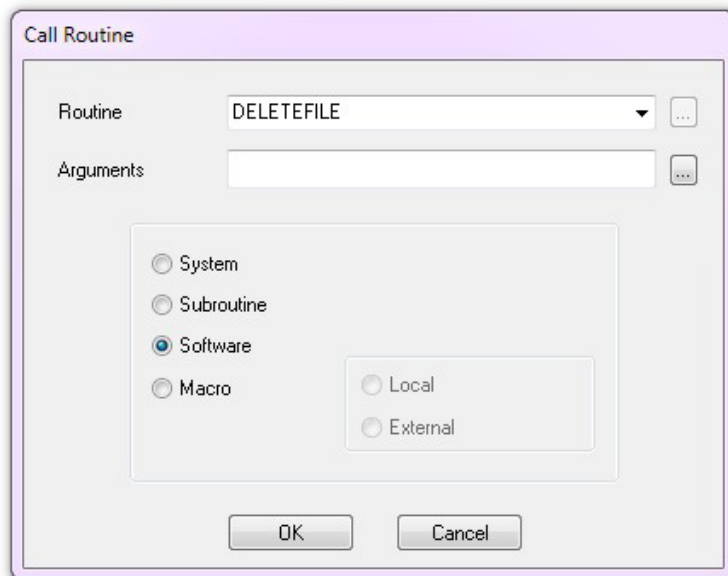


Click 'OK' to complete the process. The file will now be copied to the desired location.

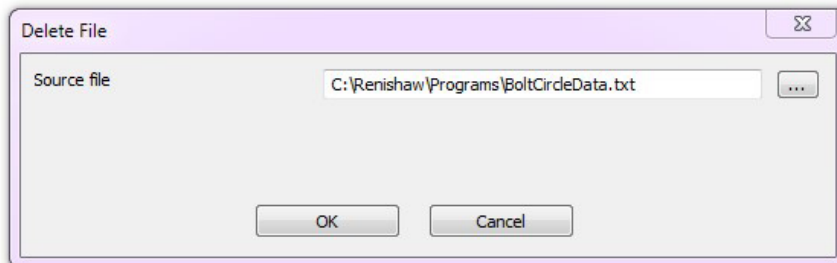
### 13.3 Delete file

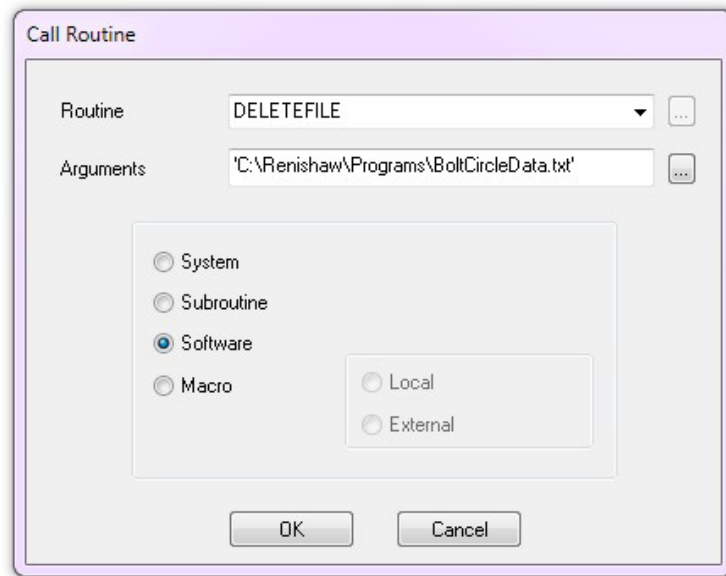
Files can also be deleted in the same way as the last two examples.

Click 'High Level' then select 'Call'. Click the 'Software' radio button and then select 'DELETEFILE' from the 'Routine' drop down menu.



Click on the 'Arguments' browse button and search for the file to be deleted. Click 'OK'.





Click 'OK' to complete the process. The file will now be deleted.

This page intentionally left blank

**Renishaw plc**  
New Mills, Wotton-under-Edge,  
Gloucestershire, GL12 8JR  
United Kingdom

**T** +44 (0)1453 524524  
**F** +44 (0)1453 524901  
**E** [uk@renishaw.com](mailto:uk@renishaw.com)  
[www.renishaw.com](http://www.renishaw.com)

**RENISHAW**   
**apply innovation™**

**For worldwide contact details,  
please visit our main web site at  
[www.renishaw.com/contact](http://www.renishaw.com/contact)**



H - 1000 - 5334 - 04